SIEMENS

# SINEC

# TF–Net 1413/ MSDOS/Windos ™

## Manual

C79000–G8976–C023

Release 02
Volume 1 of 1

## Safety Guidelines

This manual contains notices which you should observe to ensure your own personal safety, as well as to protect the product and connected equipment. These notices are highlighted in the manual by a warning triangle and are marked as follows according to the level of danger:

**Danger**

indicates that death, severe personal injury or substantial property damage **will** result if proper precautions are not taken.

**Warning**

indicates that death, severe personal injury or substantial property damage **can** result if proper precautions are not taken.

**Caution**

indicates that minor personal injury or property damage can result if proper precautions are not taken.

**Note**

draws your attention to particularly important information on the product, handling the product, or to a particular part of the documentation.

## Qualified Personnel

The device/system may only be set up and operated in conjunction with this manual.

Only **qualified personnel** should be allowed to install and work on this equipment. Qualified persons are defined as persons who are authorized to commission, to ground, and to tag circuits, equipment, and systems in accordance with established safety practices and standards.

## Correct Usage

Note the following:

**Warning**

This device and its components may only be used for the applications described in the catalog or the technical description, and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens.

This product can only function correctly and safely if it is transported, stored, set up, and installed correctly, and operated and maintained as recommended.

## Trademarks

SIMATIC and SIMATIC NET are registered trademarks of SIEMENS AG.

Third parties using for their own purposes any other names in this document which refer to trademarks might infringe upon the rights of the trademark owners.



Siemens AG
A&D
Industrial Automation Systems
Postfach 4848, D-90327 Nürnberg

**Disclaimer of Liability**

We have checked the contents of this manual for agreement with the hardware and software described. Since deviations cannot be precluded entirely, we cannot guarantee full agreement. However, the data in this manual are reviewed regularly and any necessary corrections included in subsequent editions. Suggestions for improvement are welcomed.

# Contents

# NOTES

# 1 Overview

## 1.1 System Environment and Area of Application

Personal computers (PCs) suitable for industrial applications are gaining in importance in automation engineering. One major reason for this is their high cost-effectiveness. To be useful in industry, however, PCs must be integrated in the system environment of the existing programmable logic controllers and computers, i.e. they must be able to communicate with these devices.

For applications in automation engineering, SIEMENS produces the SINEC communications system (SIEMENS NETWORK ARCHITECTURE for automation and engineering). This is a system of high-performance local area networks (LANs), for example, SINEC H1 and SINEC L2.

Within the SINEC communications world, SINEC Technological Functions (STF) are provided for communication between the control systems and computers (program-program communication) and for communication between computers themselves. STF is compatible with MMS (manufacturing message specification) and also provides an interface for the transition from SINEC H1 to SINEC L2. The STF user interface is described in Chapter 5.

Apart from the SINEC components described in this manual (communications processor CP 1413 and SINEC software modules), the following environment must be available for communication:

‰ Hardware
- Cell network SINEC H1

‰ Software
- Local network management SINEC NML V3.0 and higher or COML 1413 V1.00 and higher.

The computers with which the CP 1413 can be used must also meet the hardware and software requirements described in the following table (see also the accompanying product information).

**Hardware Requirements**

The CP 1413 can be used in conjunction with computers with operating systems marked in the table.

| **Computer type** | **CPU** | **Clock frequency MHz** | FlexOS | **MS-DOS** |
|---|---|---|---|---|
| PC 32-R | 386 | 25/33 | | * |
| PC 32-M | 486SX | 20/25 | | * |
| PC 32-T | 386 | 25/33 | | * |
| PG 730 | 386 | | * | * |
| PG 750 | 386 | | * | * |
| PG 770 | 486 | | * | * |

+ **Either the memory area D0000 or E0000 must be available since the CP 1413 requires one of these two areas. Other modules musst not use this area.**

**Software Requirements**

To operate TF-NET 1413/FlexOS, a FlexOS version of 2.2/1 or higher is required on a programming device (PG). TF-NET 1413/MS-DOS requires an AT-compatible computer with MS-DOS 3.3 or MS-DOS 5.0 on which an address area D0000-DFFFF or E0000-EFFFF is freely available.

+ **TF-Net 1413/xx cannot be operated at the same time as TF-Net 1412/xx or TF-Net 141/xx on a programming device or personal computer.**

## 1.2 Integration in the System

Figure 1.1 shows the components of TF-NET 1413 and how they are integrated in the existing communications software:

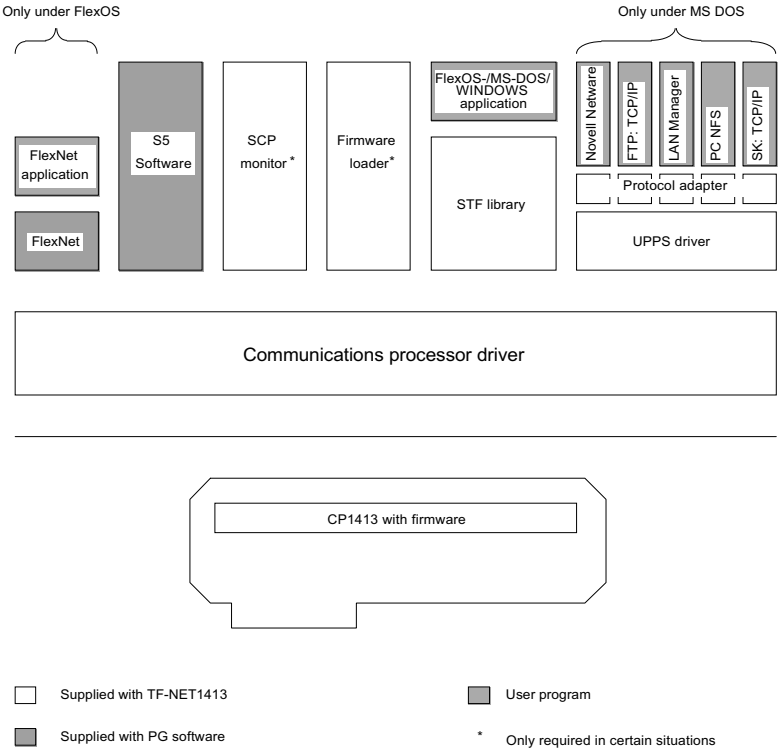



Fig. 1.1: Components of TF-NET 1413

User applications require the STF library for communication.

TF-NET 1413 consists of the following components:

‰ Communications processor driver:
The communications processor driver passes data from the user program to the CP and vice versa. It does not provide a direct interface to the user program.

‰ The SCP monitor:
The SCP monitor is used to configure the driver and handles other administrative tasks.

‰ The STF library:
The STF library contains the STF user interface for creating and processing TF-PDUs.

‰ The installation tool:
The installation tool is used to generate a configuration file which contains all the data for configuring the communications processor driver. This allows simple adaptation of the driver the existing system environment.

‰ The firmware loader:

The firmware loader copies firmware required for operating the communications processor when the computer is started up.

‰ The firmware:

The firmware on the CP 1413 handles the major part of the SINEC protocols automatically and is controlled by previously assigned parameters. On the one hand, the necessary address information (Ethernet address) for CP communication is transferred and entered in the local data base, on the other hand, parameters can also be selected to optimize the SINEC system for a wide variety of applications. The parameters specific to the computer (for example dual-port RAM address and Ethernet address) are entered using the supplied installation tool, written to the configuration file and passed on to the communications processor driver by the SCP monitor.

The network communications parameters required for STF communication (for example static application associations) are configured using NML or COML 1413 TF, written to the local data base and transferred to the CP 1413 during start-up.

User programs call the STF library functions. These use operating system calls to be able to communicate with the driver. The communications processor is addressed via I/O addresses, data exchange is via a dual port RAM. q

# 2 Hardware Installation

## 2.1 The CP 1413

The CP 1413 requires an AT slot. On the module itself, only the configuration register needs to be set using jumper X3. The position of this jumper can be seen in Figure 2.1. The default value of the configuration register is 0x03e0 (both jumpers inserted).

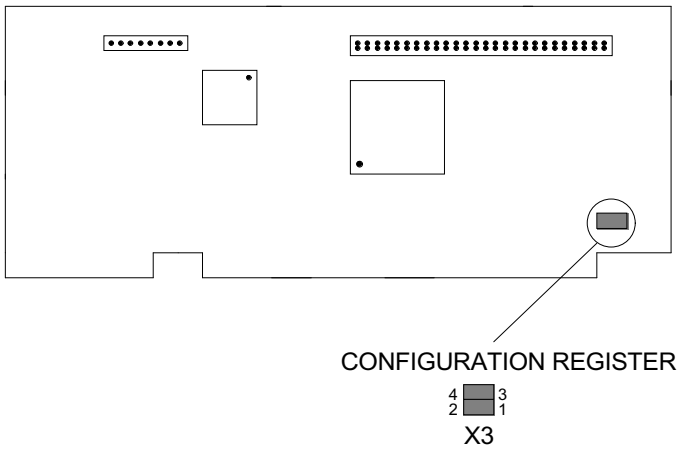



Fig. 2.1: Layout of the CP 1413

Apart from the default value, the jumper settings shown in Figure 2.2 are also possible.

| Jumper / Conf. reg. | X3 / 1-2 | X3 / 3-4 |
|---|---|---|
| 03E0H | ●—● | ●—● |
| 0390H | ●—● | ● ● |
| 03E8H | ● ● | ●—● |
| 0100H | ● ● | ● ● |

Fig. 2.2: Configuration register

All the values required for operating the module are loaded on the CP by the software via the configuration register.

## 2.2 Notes on Installation

+ **The module must only be installed in the computer when the power supply is switched off. When handling the module, the normal rules for electrostatically sensitive devices must be adhered to.**

Information about opening the device, selecting the slot for the communications processor and installing modules in this format can be found in the manual for your PG or PC. q

# 3 Software Installation

☞ **Before installing the software, read the Important Notes (Chapter 10).**

## 3.1 The Installation Tool

The installation tool is used for installation and configuration of the software and firmware required for operating the CP 1413 module. During the software installation, the system environment for TF−NET1413/MSDOS,Windows is generated. All the data of the driver and the module are written to a configuration file (\sinec\data\dos_conf.dat). The installation tool allows simple adaptation of the driver to the existing system environment.

Some of the files on the installation diskette are compressed. For this reason, the TF−NET1413 software can only be copied from diskette to hard disk using the installation tool.

The installation tool display various screens in which you are requested to make input. A value is proposed by the software for all the parameters and these can be accepted simply by pressing the return key. If you require a different value, then depending on the screen, you can make your selection in one of the following ways:

- select a window,
- enter a value within certain limits or
- type in a text (paths and file names).

You are guided through the procedure by the instructions in the menus. By pressing 'F1' , you can display help texts containing further information about individual menu items.

The proposed value for the address of the configuration register corresponds to the jumper setting of a new module direct from the factory. The configured value and the value selected with jumpers on the module must match.
The installation tool also checks whether selected values or ranges of values are permitted. The hardware characteristics of the PC are particularly important for configuring TF−NET1413/MSDOS, Windows. TF−NET1413 (the transport parameters of the CP 1413 ) is configured using the product NML or COML1413. During this configuration, a data base is generated known in later sections as the (LDB) local data base.

Figure 3.1 illustrates the procedure for first−time installation and reconfiguration.

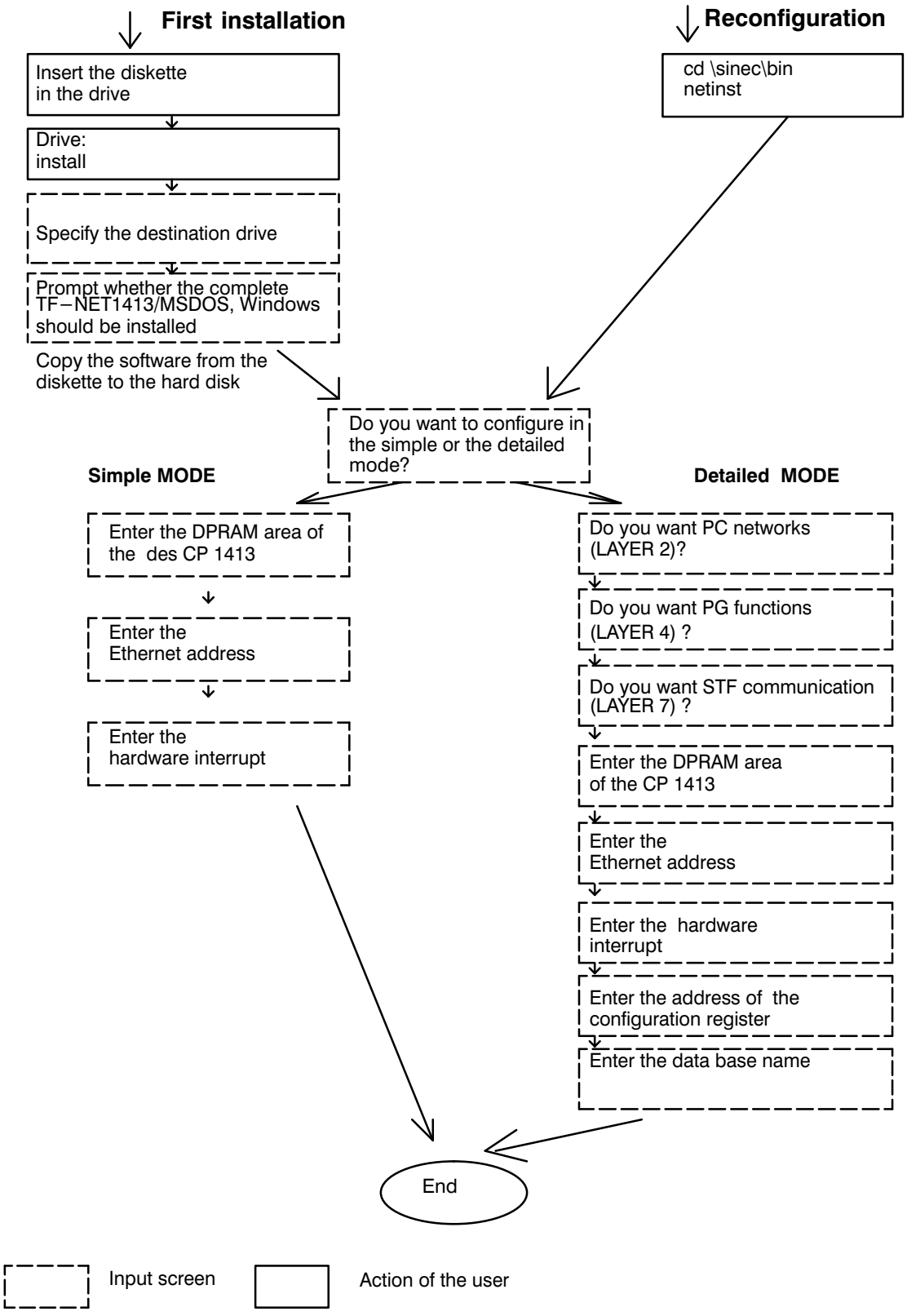



Fig. 3.1 Installation and configuration

**First Installation from Diskette**

To install the TF−NET1413/MSDOS, Windows software (simply TF−NET 1413 below), proceed as follows :

- 1) Insert the diskette in drive a: or b:
- 2) Change to the appropriate drive by typing in 'a:' or 'b:'.
- 3) Call the installation program with 'install'

The installation program netinst.exe copies itslef to the hard disk during installation so that the supplied diskette is not needed for reconfiguration. For this reason, you are asked during the first installation to name the drive on which TF−NET1413/MSDOS, Windows iis to be installed. Here, specify a drive name (e.g. C , D , E ...). The installation tool proposes drive 'C'.

The next menu asks whether you want to install the complete TF−NET1413 software or only a **subset for operating STEP5/ST2 (SIMATIC PG software, S5DOS−ST Stage 6)**. In this case, the STF software ( include files, libraries and example program soureces) are not copied to the hard disk to save space.
The complete TF−NET1413 software requires approximately 2 Mb, the subset approximately 0.8 Mb on the hard disk.

When installing TF−NET1413/MSDOS, Windows there are two modes; the simple ode and the detailed ode. In the simple mode, you are asked to specify only absolutely necessary parameters. In the detailed mode, you can change other parameters such as the I/O address of the CP 1413 (see Figure 3.1).

You select the mode in the next screen.

**The Simple Mode:**

In the first screen of the simple mode, you set the **memory area** of the 64 Kb dual−port RAM. Possible start addresses are D0000 and E0000. The selected area must not be used by any other module. If the ShadowBIOS or the cache is within this area, these must be disabled. Using the program \sinec\bin\dpramtes.exe , you can test whether the selected area is suitable for the DPRAM of the CP 1413 (The syntax of dpramtes is described n the section "Dealing with Problems" ).

☞ **As the user of a SIMATIC programming device PG 730, PG 750, and PG 770, you must consider the following points:**

**in the programming devices, the area E0000 is already occupied by the installed ARCNET interface module. For TF−NET1413, select the area D0000. This allows simultaneous operation of ARCNET and TF−NET 1413.**
**If you want to operate TF−NET1413 (SINEC H1) alongside TF−NET5412 or TF−NET5410B (SINEC L2), you must assign the dual−port RAM addresses as follows:**

**The SINEC−L2 module (CP 5410B or CP 5412) has the address D0000. This module is activated at the AT bus of the PG immediately after the power supply is switched on and it cannot be disabled.**
**In contrast to this, the SINEC H1 module only switches n when the driver is loaded. ARCNET is activated by assigning an ARCNET node address in the Set-up menu. This means that the two interface modules can use the same DPRAM area one after the other.**

**If you have installed all three communications types (SINEC H1, SINEC L2 and ARCNET), select the address E0000 for TF−NET 1413.**

**Communication via ARCNET with three communications processors installed:**

- **1. Enter a valid ARCNET address in Setup.**
- **2. TF−NET 1413 must not be started.**

**Communication via SINEC−H1 with three installed communications processors:**

- **1.Enter the ARCNET address 0 (not installed) in the Setup.**
- **2. Start TF−NET 1413.**

**SINEC L2 is operable in both cases.**

| Simultaneous operation | **ARCNET** | **CP 1413** | **CP 5410B or CP 5412** |
|---|---|---|---|
| **ARCNET** | | Valid ARCNET address in Setup<br>CP 1413 at DPRAM address D0000<br>No L2 module plugged in | Valid ARCNET address in Setup<br>CP 5412 at DPRAM address D0000<br>CP 1413 at E0000 but the driver has not started |
| **CP 1413** | | | ARCNET address 0 in SETUP<br>CP 5412 at DPRAM address D0000<br>CP 1413 at DPRAM address E0000 |

**Fig. 3.2 Simultaneous operation of communications modules in the PG**

In The second screen, you are asked for the **Ethernet address** to be set on the CP 1413. Providing **no** data base (icalled the (LDB) local data base or simply data base below) generated by the configuration tool exists. If an (LDB) local data base does exist, the Ethernet address is overwritten by the Ethernet address of the data base.

The CP 1413 requires a hardware interrupt to indicate an event to the PC. The following interrrupt numbers are allowed:

5
10 (default)
12
15

☞ **The selected hardware interrupt must not be used by any other module.**

The following values are fixed in the simple mode :

| | | | | |
|---|---|---|---|---|
| Configuration register (IO address) of the CP 1413 : | 3E0 | | | |
| Data base name: | \sinec\data\startup.ldb | | | |
| CP1413 channels: | PC networks (Layer 2) | + | PG functions (Layer 4) | + STF applications (Layer 7) |

If you want to change any of the fixed parameters, use the detailed mode.

**The Detailed Mode:**

In the first three screens of the detailed mode, you are asked whether you want communicaation via LAYER 2 , LAYER 4 or LAYER 7. You require LAYER 2, if want to use one of the protocols via UPPS, e.g. Novell, TCP/IP, NFS, LAN manager. You require LAYER 4 if you want to operate MSNET and the SIMATIC PG software (STEP5/ST2 ). LAYER 7 is required if you want to use STF applications. Several channels can be configured. Each channel, however, requires resources in the DP RAM. In terms of tuning, you may find it advisable to configure only the channels that are actually needed. If you want to use a channel, select 'YES', otherwise 'NO'.

In the following screens, just as in the simple mode, you are asked to specify the DP RAM area, the Ethernet address and the number of the hardware interrupt (for more detailed information on these points refer to the 'simple mode'.

The CP 1413 is addressed via a configuration register. The I/O address of this configuration register is set using a jumper on the CP 1413. The default is 3E0. If this value is already occupied, you must modify the jumper setting on the CP 1413. This value must then be entered in the next screen.

The following values are possible :

3E0 (default)
390
3E8
100

**The hardware configuration address of the CP 1413 must match this address..**

The communications parameters of the CP 1413 are written to the data base which is loaded on the module when you start the TF−NET 1413 software. The default menu of this data base is \sinec\data\startup.ldb. You can change this name in the next screen.

The installation of TF−NET1413/MSDOS, Windows is then complete

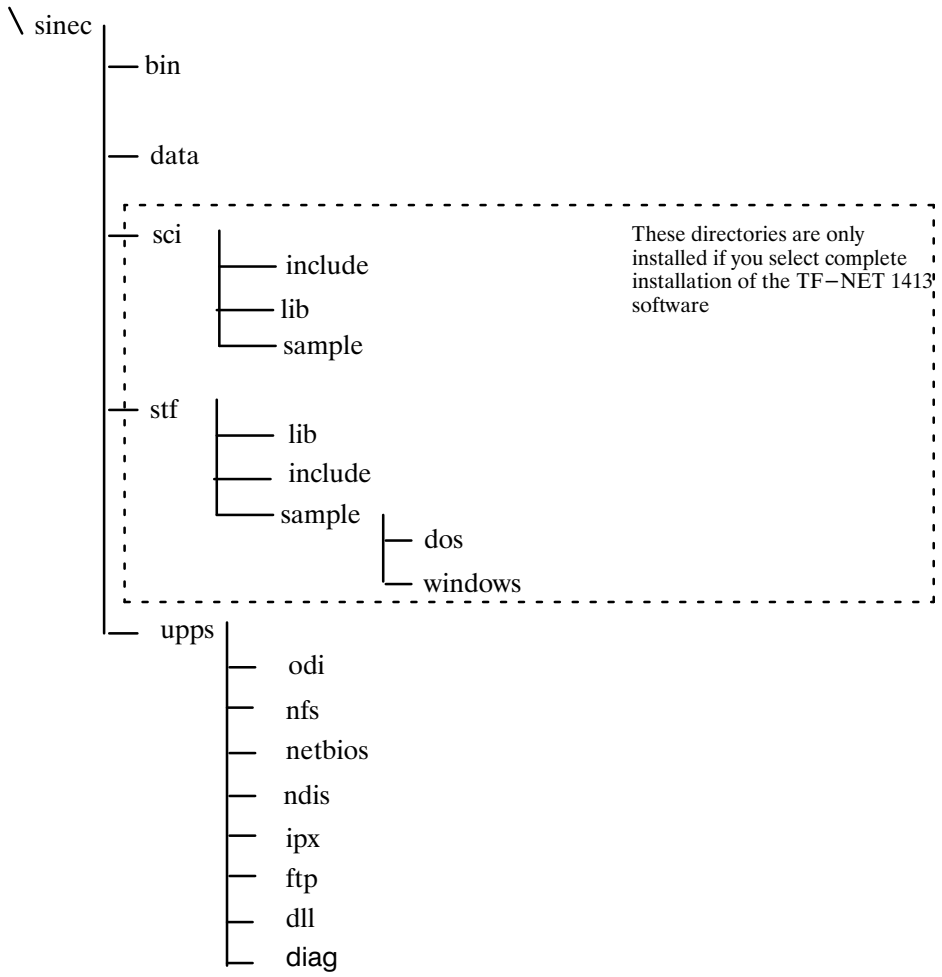
**Reconfiguring from the hard disk**
After the first installation of the TF−NET 1413 software from the diskette, you can make modifications to the system environment from the hard disk.

- Change to the directory in which the program ”netinst.exe” is located:
  cd \sinec\bin
- Call the program ”netinst.exe”:
  netinst
- Follow the menu−guided instructions (see ”simple mode” and ”detailled mode”).

Reset the computer (cold restart, hard reset).

**Directory structure**

When you install from diskette, the following subdirectory structure is generated on the hard disk:




During installation, the files of the driver are copied to the directory \sinec\bin. The subdirectory '\sinec\data' contains all the files generated during installation.

'DOS_CONF.DAT':

This contains the data generated during installation for the SCP monitor and the driver.

'DOS_CONF.TXT':

All the input made during configuration is logged in this file.

## 3.2 Starting the Module and the Driver

✓ To start the driver, change to the directory \sinec\bin and start the script startcp.bat :

```
cd \sinec\bin
startcp
```

The batch file startcp.bat starts the driver (tfnetdrv.exe), loads the firmware on the module and starts it (scp_mon −c ). The data base created with NML or COML 1413 is then loaded on the module (scp_mon −l ). Following this, layer 4 or layer 7 applications can be started. Starting layer 2 applications such as the Novell Netware, PC−NFS .... is described in the chapter ”UPPS”.

Starting the communications driver:

```
cd \sinec\bin
tfnetdrv
scp_mon −c
scp_mon −l
```

These commands, which are located in startcp.bat, can also be entered in the autoexec.bat file. The driver is then started when MS−DOS is started.

☞ As standard, the driver uses the software interrupts 0x80 and 0x81. If one of these software interrupts is already being used by another driver, you can set the shell variables **SCI_SW_INT** before starting the driver to force the use of a different software interrupt. Two consecutive software interrupts are always required:

Example: set SCI_SW_INT=82

In this case, TF−NET1413 uses the software interrupts 0x82 and 0x83 .

If the software interrupts 0x80 and 0x81 are being used and the shell variable SCI_SW_INT does not exist, the driver itself searches for a free interrupt in the area 0x82 to 0xe6.

## 3.3 The SCP Monitor

The data required for configuring the TF−NET1413 software (particularly the interface driver, the module and the dual−port RAM) are located in the directory \sinec\data and the programs such as the SCP monitor in the directory \sinec\bin . You must therefore start the scp_mon on the drive on which TF−NET 1413 is installed.
Among other things, the monitor has the task of transferring the configuration data to the driver so that this can configure both the dual−port RAM and the module. For each module, the firmware is loaded and started so that the station is ready for communication and user programs can use the

driver and CP.

Call syntax : scp_mon <option>

The following values are possible for the <option>

| | |
|---|---|
| −a | The local names of all applications associations and server names configured on the CP 1413 are read out and displayed on the screen. |
| −c | The firmware is loaded on the module, the driver configured and the module started. |
| −h | The transport connections are cleared. The CP 1413 is reset and the driver removes itself from the main memory of the PC. |
| −l | The data base is loaded on the module. |
| −p | The buffer statistics (buffer bottlenecks, buffer requests and the number of buffers) is output. |
| −r | The CP 1413 is reset by a software reset. The firmware does not need to be reloaded on the module. Following this, you can load a data base on the CP 1413 with scp_mon −l . |
| −rf | There is a hardware reset on the CP. The driver remains in the memory. You can start the TF−NET1413 software again with scp_mon−c and scp_mon−l. |
| −s | The SC monitor outputs the status of the local data base of the module. The system manager can then check whether a data base loaded by the host or the standard data base is active. |
| −u | The SCP monitor saves the local data base of the module on the hard disk of the host computer. The file in which the data base is written is the file whose name was specified during configuration in the detailed mode. As standard, this file is \sinec\data\startup.ldb. The old file is first stored in a file with the extension .old. |
| −v | The current release of the SCP module is read out. |
| −? | A description of the scp_mon parameters is displayed on the screen. |

## 3.4 The Driver

The installation of the communications processor driver requires changes both in the hardware and software configuration of the system.
The modules must be integrated according to the existing hardware configuration. This depends on the other modules currently installed.
The data required for assigning parameters to the driver, the dual−port RAM and the modules are stored in a configuration file with the installation tool already described above. When installing the driver, internal data structures are set up and some are initialized. The further initialization of the driver is then performed by special driver calls using the scp_mon command (SCP monitor).

As standard, the driver uses the software interrupts 0x80 and 0x81. This default can, however, be modified by an environment variable (see Section 3.2).

## 3.5 Configuration

Using the NML product (Network Management for LAN, see /1/ ) or COML 1413, you generate a data base with configured connections. Once you have generated the data base, there are two ways in which you can inform the TF−NET1413 software of this data base.

1) ✓ Copy the data base to diskette.
   ✓ Copy the data base from diskette to the hard disk of the destination computer. The file name must match the name selected during TF−NET1413 configuration (default: \sinec\data\startup.ldb).

2) ✓ Load the data base on the destination PC via the LAN. This data base is then immediately valid on the CP 1413.
   ✓ Save the data base on the hard disk with the command:
   scp_mon −u
   or when shutting down TF−NET1413 with:
   scp_mon −h.

(Note: with scp_mon −h, the data base is only saved on the hard disk when a data base was loaded on CP 1413 using NML or COML 1413. With scp_mon −u, the data base on the module is always saved on hard disk. The old data base is saved in a file with the same name but with the extension .old,

## 3.6 The UPPS Interface

UPPS (universal portable protocol stack) is a layer 2 interface independent of the protocol which uses the communications processor driver. Its use with protocols of different manufacturers, for example Novell and Sun is achieved using adapter modules. This section shows how TF_NET1413/MSDOS, Windows can be integrated in the various protocols using UPPS.

**When using the following interfaces, TF−NET1413 must be started (startcp : tfnetdrv, scp_mon −c , scp_mon −l) as well as the UPPSDLI driver c1413dli.com.** The integration of these adapter modules in the network software of the products is described in the individual sections.

### 3.6.1 ODI : NOVELL Netware 3.x , NOVELL Lite

UPPSODIE.COM is the Multiple Link Interface Driver (MLID) for the Open Datalink Interface (ODI) from Novell within the framework of UPPS and is located in the directory \sinec\upps\odi. Information about installing Novell netware can be found in the original documentation from Novell.

Before starting UPPSODIE.COM, you must load the Data Link Interface (c1413dli.COM) and the Link Support Layer (LSL.COM).

Following this, you can load the protocol driver IPXODI.COM.
LSL.COM and IPXODI.COM are on the Novell Netware workstation diskette or on the Novell Lite installation diskette.

The configuration file NET.CFG must be located in the current directory in which IPXODI.COM is started. The parameters for UPPSODIE are described in the section

Link Driver UPPSODIE

For further information about the structure and content of NET.CFG, refer to the ODI documentation from Novell.

An example of the file net.cfg for communication with a Novell 3.0 fileserver can be found in the directory \sinec\upps\odi:

```
Link Driver UPPSODIE
        Frame Ethernet_II
        Protocol IPX 8137 Ethernet_II
```

IPXODI.COM is set to the Ethernet type code 8137 (hex) registered for Novell IPX.

If you want to activate the Novell Netware Client when starting your PC, enter the commands listed below in the \autoexec.bat file.

```
cd \sinec\bin
tfnetdrv
scp_mon −c
scp_mon −l
c1413dli
cd   <novell_dir>
lsl
cd \sinec\upps\odi
UPPSODIE
cd   <novell_dir>
IPXODI
NETx
```

<novell_dir> stands for the directory in which the Novell Client Software is installed. NETx stands for the netware shell responsible for your DOS version, i.e. NET3.COM, NET4.COM or NETX.COM for MSDOS 5.0.

Note
All the programs in the example above can be removed completely from the memory again. To do this, the programs must be called in the reverse order with the appropriate parameter:

```
NETx −u
IPXODI −u
UPPSODIE −u
LSL −u
C1413DLI  −u
```

### 3.6.2 IPX : NOVELL Netware 3.x , NOVELL Lite

Instead of the ODI interface (see Section 3.5.1), the Novell Netware Client and the Novell Lite Software can also be operated with the IPX interface.

The following files must be located in the directory \sinec\upps\ipx:

| | |
|---|---|
| − UPPSIPX.OBJ | Object file for generating the shell |
| − UPPSIPX.LAN | LAN description file |
| − UPPSIPX.COM | Generated shell (version 3.01 Rev A) |
| − UPPSIPX.DOC | |

UPPSIPX.COM is the protocol driver for IPX/SPX from Novell within the framework of UPPS.

Before starting UPPSIPX.COM, you must load the data link interface (C1413DLI.COM)

UPPSIPX.COM must be configured with the Novell command ECONFIG.EXE for a particular Ethernet type code or for IEEE 802.3.

After starting UPPSIPX.COM, the netWare shell (NET3.COM, NET4.COM or NET5.COM) is started and you can log on at a netware file server.

If you want to activate the Novell Netware Client when you start your PC, enter the following lines in the \autoexec.bat ein.

```
cd \sinec\bin
tfnetdrv
scp_mon −c
scp_mon −l
c1413dli
cd \sinec\upps\ipx
uppsipx
cd <novell_dir>
NETx
```

<novell_dir> stands for the directory in the Novell Client Software is installed. NETx stands for the netware shell responsible for your DOS version, i.e. NET3.COM, NET4.COM or NETX.COM for an MSDOS 5.0.

UPPSIPX.COM implements the Novell IPX version 3.02 (can therefore be used with the newest shell) and supports DOS 5.0, MSDOS 3.x , Windows 3.0 and Windows 3.1.

If you nevertheless want to generate your own IPX.COM with the Novell command SHGEN (Generate Shell) proceed as follows:

- Format a diskette and assign the volume label LAN_DRV_215.
- Copy the object file UPPSIPX.OBJ and the LAN description file UPPSIPX.LAN to this diskette.
- Start SHGEN
- Select:

  Load and Select

  to generate IPX.COM.

Generate the shell (SHGEN) from the hard disk or file server:

- Copy the object file UPPSIPX.OBJ and the LAN description file UPPSIPX.LAN to the subdirectory LAN_DRV_.215.
- Start SHGEN
- Select:

  Load and Select

  to generate IPX.COM.

**There is, however, already a working version of UPPSIPX.COM in the directory \sinec\upps\ipx.**

Note: Configure UPPSIPX.COM (with ECONFIG) with the official type code 8137 (hex). The supplied UPPSIPX is already set to this type code.
Set all the IPX components in your network (file server, other clients...) to the now official Novell type code 8137 (hex).

### 3.6.3 PC−NFS

UPPSNFS.SYS is the link level driver for PC−NFS from Sun Microsystems within the framework of UPPS. UPPSNFS.SYS is located in the directory \sinec\upps\nfs.

UPPSNFS.SYS is a DOS device driver.

- Include the following in the CONFIG.SYS file:
  DEVICE=\sinec\upps\nfs\UPPSNFS.SYS [/sxxxx]

  With /s, you specify the maximum packet size on the Ethernet network. The value xxxx is specified as a decimal number. As standard, the value 1514 is used. This parameter is necessary if Novell Netware 386 is used in the network as a router or as an NFS server. You must set the Novell Server in startup.cnf with the parameter ”maximum physical receive packet size”.
- Enter the device drivers PCNFS.SYS supplied with PC−NFS and SOCKDRV.SYS in the CONFIG.SYS file.
- Reset your computer so that the drivers entered in CONFIG.SYS are loaded.
- Then start the CP 1413 communications processor and the data link interface (C1413DLI.COM) .
- Following this, start the PC−NFS services (for example using the PC−NFS program NET.EXE) .

For further information about operating PC−NFS, refer to the documentation supplied with PC−NFS.

Example of the files \config.sys and \autoexec.bat:

config.sys:

```
DEVICE=\NFS\PCNFS.SYS
DEVICE=\NFS\SOCKDRV.SYS
DEVICE=\SINEC\UPPS\NFS\UPPSNFS.SYS
```

autoexec.bat:

```
cd \sinec\bin
tfnetdrv
scp_mon −c
scp_mon −I
c1413dli
net .....
```

### 3.6.4 NETBIOS

UPPSNB.EXE is the NetBIOS Emulator (host−based NetBIOS) within the framework of UPPS.

- Start the CP 1413 communications processor driver
- Start \sinec\bin\c1413dli.com.
- Start \sinec\upps\netbios\uppsnb.exe.

If you want to activate the NetBIOS Emulator when you start your PC, enter the following lines in the \autoexec.bat file.

```
cd \sinec\bin
tfnetdrv
scp_mon −c
scp_mon −I
c1413dli
cd \sinec\upps\netbios
uppsnb
```

**Command line parameters**

You can specify the following parameters for calling UPPSNB :

```
−s  maximum number of simultaneous sessions, default=6
−n  maximum number of pending commands (NCBs), default=12
−m  maximum number of local NetBIOS names, default=12
−b  number of data buffers, default=8
−u  remove UPPSNB from memory
```

### 3.6.5 NDIS, LAN Manager

UPPSMAC.DOS is a Media Access Control (MAC) driver according to the Network Driver Interface Specification (NDIS) 2.0.1.

UPPSMAC.DOS is a DOS device driver and must be entered in CONFIG.SYS as follows:

```
DEVICE=\sinec\upps\ndis\UPPSMAC.DOS
```

The protocol manager (PROTMAN.DOS) must already be loaded.

The device name of the driver is UPPSMAC$. The module name of the driver is UPPSMAC. This must be specified in the PROTOCOL.INI file as follows:

```
[UPPSMAC]

        DriverName = "UPPSMAC$"
```

Before calling NETBIND.EXE, the data link interface (C1413DLI.COM) must be loaded.

Example:

To allow access to a LAN manager file server containing the TCP/IP protocol, the following lines must be included in the PROTOCOL.INI file:

```
[ProtMan]

        DriverName =        "PROTMAN$"

[UPPSMAC]

        DriverName =        "UPPSMAC$"

[TCPIP_XIF]

        DriverName =        "TCPIP$"
        Bindings   =        "UPPSMAC"
```

The CONFIG.SYS file must contain the following lines:

```
DEVICE=PROTMAN.DOS
DEVICE=UPPSMAC.DOS
DEVICE=TCPIP.DOS
```

The AUTOEXEC.BAT file must contain the following lines:

```
cd \sinec\bin
trei1413
scp_mon -c
scp_mon -l
C1413DLI
NETBIND
```

When operating the LAN manager via TCP/IP, the name TCPIP in the files above must be replaced by NETBUI.

For example: Protocol.ini :

```
[NetBEUI_XIF]

        DriverName =    "NETBEUI$"
        Bindings   =    "UPPSMAC"
```

config.sys:

```
DEVICE=NETBEUI.DOS
```

### 3.6.6 FTP TCP−IP

UPPSPD.COM is the FTP packet driver within the framework of UPPS. It corresponds to the PC/TCP 1.09 (basic + extended) packet driver specification of FTP.

- Start the CP 1413 communications processor driver.
- Start \sinec\bin\c1413dli.com.
- Start \sinec\upps\ftp\uppspd.com .

UPPSPD has no command line parameters.

If you want to activate FTP TCP/IP when you start your PC, enter the following lines in the \autoexec.bat file:

```
cd \sinec\bin
tfnetdrv
scp_mon −c
scp_mon −l
c1413dli
cd \sinec\upps\ftp
uppspd
```

After starting UPPSPD, you can load the TCP/IP kernel ETHDRV.EXE of FTP and/or appropriate programs. (e.g. Telnet−Emulation, LANWatch).

❑

## NOTES

# 4 STF under MSDOS and Windows

## 4.1 General

The STF interface is made available to the user in the form of libraries. the library modules of the STF interface handle the setting up and processing of TF−PDUs for the user and the service−oriented performance of jobs or acknowledgements. The services of the STF user interface can be divided into the following:

1) Administrative services required for initializing STF applications.

2) Non−open services, to ensure compatability with existing systems, such as SIMATIC S5 with the CP 535. These services cannot be modelled on MMS protocols.

3) Open services which can be modelled on MAP 3.0 MMS (IS 9506).

The SINEC−TF user interface is described in Chapter 5.

The STF libraries are supplied for the **'Large' memory module** and installed in the directory \sinec\stf\lib. Libraries are available both for DOS and Windows for various compilers.

With STF applications, the module with the name "SCP" (SINEC Communications Processor) is addressed (default). This name is specified in the stf_conf.dat file.

### 4.1.1 Header Files

The header files are located in the directory \sinec\stf\include.

The header file **stf.h** is the standard STF header file required by all STF applications. This header file contains information and definitions for the SINEC technological functions. stf.h itself includes the header file **stf1.h** .

The header file **htb.h** is required for using handling block functions.

The header file **stf_func.h** is important for integrating various STF services in the application program. The actual content of this header file is of no interest to the user and must not be modified. This header file must not be integrated in the user program **more than once**.

The stf_prty.h file contains the protocol definitions of the STF library functions. To activate the prototype check, include the following lines in your program:

```
#define TE_PROTO
#include "stf.h"
#include "stf_prty.h"
```

### 4.1.2 STF Library

The STF library contains all the functions of the STF user interface for modules. Each user application includes the library modules in its program automatically.

### 4.1.3 Configuration File

Using the configuration file **stf_conf.dat** the programmer can adapt the STF library to his own needs. The configuration file **must be in the same path** as the executable program since it is read when the STF library is initialized (i.e. tf_init). Based on the parameters, the STF library requests dynamic memory. If this file does not exist, the parameters specified in the following example are used (refer to the example of the configuration file in /usr/stf/example/stf_conf.dat).

Most of the parameters with dimensions in the configuration file are returned to the user by the function *tf_init* with the structure DIM_PARAM.

```
/*********************************************************************/
/*                  Copyright  (C)  Siemens AG 1991  All Rights Reserved          */
/*********************************************************************/
/*  PROJECT:                    STF Interface                          */
/*  TITLE:              Configuration File                             */
/* VERSION:                     initial version for default values     */
/*  FILE NAME:          stf_conf.dat                                   */
/* SYSTEM:              MSDOS / WINDOWS                                */
/*  Last Modification:  30.10.1991                                     */
/*                                                                     */
/*********************************************************************/

SCP_device          = SCP           /* SCP device path name           */
auto_rsp_conc  = TRUE               /* automatic response for conclude! */
auto_dom_serv = TRUE                /* automatic domain server capability!*/
with_var_spec       = TRUE          /* specification_with_result (tf_read)  */
max_mess_recv= 1            /* max. messages to be received once      */
                            /* (only 1 possible)                      */
max_appl_rel        = 10            /* max. application relations     */
max_server_id       = 3             /* max. server identifications    */
max_download= 1             /* max. simultaneous download             */
max_upload          = 1             /* max. simultaneous upload       */
shareable   = TRUE          /* this default value is used only if     */
                            /* additional_info is not specified!      */

/*********************************************************************************/
/*                  Copyright  (C)  Siemens AG 1991  All Rights Reserved          */
/*********************************************************************************/
```

☞ **The parameter SCP_device selects the required CP 1413 module via which communication will be handled. The current version of TFNET1413/ MSDOS only supports one board. For this reason, this parameter must not be changed in this version.**

With the parameter *auto_rsp_conc = TRUE* the *tf_rsp_conclude* function is automatically processed in **tf_receive**.

Using the parameter *auto_dom_serv = TRUE* the events request domain download indication and request domain upload indication are handled automatically by performing the functions tf_download or tf_upload in tf_receive on the file server (domain server).

The parameters *with_var_spec* and *shareable* are STF protocol options which can be controlled by the user. With *with_var_spec = TRUE* the variable specification is returned in the acknowledgement. If this option is not required or if a server requires the value FALSE, this parameter can be modified.

The remaining parameters are required for memory allocation within the STF library. For example, the default for the maximum number of servers to be logged on is three. If you want to log on more than three servers, you must enter the required value in the line "max_server_id = ".

☞ **The value for max_mess_recv (number of messages that can be fetched with a tf_receive call) is one and must not be modified.**

### 4.1.4 Compilation Options

The STF library contains all the services of the SINEC technological functions. To avoid having to have all the STF services integrated in the user program, for example to save memory, there are compilation definitions for specific services to be able to deselect STF services.

The following compilation options are available:

**NO_TIM** to deselect the **time services**,

**NO_SER** to deselect the **serial transfer**,

**NO_VAR** to deselect the **variable services**,

**NO_DOM** to deselect the **domain services**,

**NO_PI** to deselect the **program invocation services**

These services are then not integrated in the user program. Define the option when translating.

Example of a confirmation without domain or PI services (with the MSC compiler ):

```
cl ..... −DNO_PI  −DNO_DOM
```

### 4.1.5 Byte Alignment

Normally, variables are stored in memory by the compiler in the form which appears to be the most sensible for the compiler. Between components of variable, gaps can occur (padding bytes). The TF library, however, assumes that there are no gaps (byte alignment) in the variables. All the variable structures must therefore be parenthesized with **#pragma pack(1)** and **#pragma pack()** (refer to the program example in Chapter 7).

### 4.1.6 Length Restrictions for Messages

All STF calls are subject to the restriction that the message (i.e. including the AP header) must not be **longer than 4000 bytes**.

### 4.1.7 Length Restriction for Handling Block Calls (CP 143, CP 535)

The "*permitted range*" of the source/destination block depends on the set or configured **max. TIDU size** (TIDU = Transport Interace Data Unit) in the local data base (see NML description):

a) in words amount <= (max_TIDU_Size − 16)/2
b) in bytes amount <= (max_TIDU_Size − 16)

The maximum amount, however, must not exceed **2000 words** or **4000 bytes**.

Parameters must be assigned to the CP 1413 as described in the appropriate manual.
The maximum TIDU size must be set in the transport connection profile.

### 4.1.8 Defines

With the tf_receive call, the element tf_service is returned as part of the structure pb_recv. The values to be returned are defined in stf.h. The names WRITE and READ were intended for the variable services. These definitions, however, already exist with other values in header files of Compilers, e.g. in windows.h ... . For this reason, the tf_service definitions for the variable services are VAR_READ and VAR_WRITE.

## 4.2 Synchronous Function Calls

The STF user interface provides synchronous and asynchronous calls. A synchronous call means that the call only returns to the caller on completion of the job. The mode is controlled in most STF functions using the **mode** parameter (mode == CONF_SYNC : synchronous mode, mode == CONF_ASYNC : asynchronous mode),

☞ **While a synchronous job is active, all indications ( IND_CONF and IND_UNCONF) are acknowledged negatively until the job confirmation has arrived and the job is completed.**

**Synchronous and asynchronous calls should not be mixed. If there is simultaneous client/server operation in a program, only the asynchronous mode should be used.**

## 4.3 Asynchronous Modes

The STF user interface provides both synchronous and asynchronous calls. This allows the user to decide when it wishes to receive and process the acknowledgement for an initiated job. With the STF library there are two (for Windows three) ways of waiting for the completion of asynchronous STF function calls:

1) Simple synchronous waiting

2) Polling

With **Windows** there is a further possibility:

3) Waiting at the central loop (GetMessage ()).

With all asynchronous STF library function calls, the user can only send as many jobs asynchronously, without a **tf_receive** as allowed by the send credit. The current send credit is obtained by the user when each application association is logged on with tf_get_path in the APPL_PATH structure as the component **usr_snd_crd**. A description of configuring send credits can be found in the NML documentation.

☞ **If the send credit is ignored, calls can be rejected due to an error!**

With all asynchronous STF library function calls, the user variables and their object description and the job parameter block to be read or written via STF must be declared as **global variables**.

### 4.3.1 Single Wait

The simplest way in which a process waits for an asynchronous event or function call is to use the STF call tf_receive. When tf_receive is used, the user process is blocked until an event occurs or there is a timeout. This function is described in detail in Chapter 5 ”TF User Interface”.

### 4.3.2 Polling

With the value wait_timeout = 0 for the tf_receive call, it is possible to check whether a message has arrived or not. The call is returned immediately. If a message existed, the value of num_mess is 1, otherwise 0.

### 4.3.3 Message Loop under Windows

When a message is received, the task sin_serv.exe sends a WM_SINEC Message to the STF user program. This can then fetch the message in its WndProc Routine with a tf_receive with wait_timeout = 0. For more detailed information refer to the Section ”Special Features of Windows”.

### 4.4 Asynchronous Administrative Functions

In addition to the synchronous calls, tf_open_path and tf_close_path for maintaining dynamic application associations and tf_state_path for requesting the statuses of application associations, corresponding asynchronous calls are also available: tf_aopen_path, tf_aclose_path and tf_astate_path.

#### 4.4.1 Structure of a Dynamic Application Association

With the **tf_aopen_path** call, the dynamic application association is established explicitly. The reference obtained with tf_get_path_ref must be specified. The status of the application association is coded in the parameter block OPB_ADMIN.

You can choose between a synchronous or asynchronous call. In the synchronous mode (CONF_SYNC), the user process remains blocked until the call is completed.

In the asynchronous mode (CONF_ASYNC) the user process continues to run immediately after accepting the call. The event can be fetched at a later point in time using the tf_receive function.

In both cases, the current status of the application association is indicated.

```
INT32     tf_aopen_path ( applref,
                          mode,
                          orderid,
                          ord_timeout,
                          opb_ptr
                        );

          TYP_APPLREF       applref;
          CHAR              mode;
          UNSIG32           orderid;
          UNSIG32           ord_timeout;
          struct  OPB_ADMIN *opb_ptr;

struct    OPB_ADMIN
          {
          UNSIG16           adm_errorid;              R
          UNSIG16           status;                   R
          UNSIG32           reserved;                 I
          }
```

**Function call:**

```
result = tf_aopen_path ( applref, mode, orderid, ord_timeout,
                         &opb_aopen_block);
```

**Description of the parameters :**

- **Function call**

| | | |
|---|---|---|
| applref | Reference of the application association via which the job is to be sent. | |
| mode | CONF_SYNC | job with synchronous confirmation |
| | CONF_ASYNC | job with asynchronous confirmation |
| orderid | Job ID assigned by user. This ID can be used with asynchronous jobs as an option using r_orderid of the RCV_BLOCK to identify the acknowledgement. | |
| ord_timeout | Management time for the job (in seconds). | |
| *opb_ptr | Pointer to the job parameter block OPB_ADMIN. | |

- **Job parameter block OPB_ADMIN**

adm_errorid — Return parameter in the OPB; errors for the specific job are returned in the adm_errorid parameter.

status — This return value indicates the status of the application association. The following statuses are possible:

| | |
|---|---|
| E4V_UNKNOWN | connection not known |
| E4V_DOWN | connection not established |
| E4V_ESTABLISH | connection being established |
| E4V_UP | connection is established |
| E4V_RECOVERY | connection has broken down |

reserved — This parameter is used internally.

**Values of the result :**

The following values (high word/low word) can occur :

| | |
|---|---|
| STF_OK | No error; adm_errorid = F_OK. |
| E_OPEN_PATH/E_PARAM | Error in one of the call parameters. Either an error ID is set or the result of the access was not set by the sender module. |
| E_OPEN_PATH/E_NOINIT | The user has not called the function tf_init before the call or the tf_init was not successful. |
| E_OPEN_PATH/E_ASYNC | The mode = CONF_SYNC was selected, i.e. asynchronous procedure although an asynchronous job is still active. The job was not sent. The sender must wait until the end of the asynchronous job. |

### 4.4.2 Terminating a Dynamic Application Association

With the tf_aclose_path call the dynamic application association is explicitly terminated. The reference obtained with tf_get_path_ref must be specified. The status of the application association is coded in the OPB_ADMIN parameter block.

You can choose between a synchronous or asynchronous call. In the synchronous mode (CONF_SYNC) the user process remains blocked until the call is completed.

In the asynchronous mode (CONF_ASYNC) the user process continues immediately after the call has been accepted. The result can be fetched using the tf_receive function at a later point in time.

In both cases, the current status of the application association is included.

```
INT32     tf_aclose_path ( applref,
                           mode,
                           orderid,
                           ord_timeout,
                           opb_ptr
                         );

          TYP_APPLREF       applref;
          CHAR              mode;
          UNSIG32           orderid;
          UNSIG32           ord_timeout;
          struct  OPB_ADMIN *opb_ptr;


struct    OPB_ADMIN
          {
          UNSIG16           adm_errorid;              R
          UNSIG16           status;                   R
          UNSIG32           reserved;                 I
          }
```

**Function call**

```
result = tf_aclose_path ( applref, mode, orderid, ord_timeout,
                                    &opb_aclose_block);
```

**Description of the parameters :**

- **Function call**

| | | |
|---|---|---|
| applref | Reference of the application association via which the job is to be sent. | |
| mode | CONF_SYNC | job with synchronous confirmation |
| | CONF_ASYNC | job with asynchronous confirmation |
| orderid | Job ID assigned by user. This ID can be used with asynchronous jobs as an option using r_orderid of the RCV_BLOCK to identify the acknowledgement. | |
| ord_timeout | Management time for the job (in seconds). | |
| *opb_ptr | Pointer to the job parameter block OPB_ADMIN. | |

- **Job parameter block OPB_ADMIN**

| | | |
|---|---|---|
| adm_errorid | Return parameter in the OPB; errors for the specific job are returned in the adm_errorid parameter. | |
| status | This return value indicates the status of the application association. The following statuses are possible: | |
| | E4V_UNKNOWN | connection not known |
| | E4V_DOWN | connection not established |
| | E4V_ESTABLISH | connection being established |
| | E4V_UP | connection is established |
| | E4V_RECOVERY | connection has broken down |
| reserved | This parameter is used internally. | |

**Values of the result :**

The following values (high word/low word) can occur :

| | |
|---|---|
| STF_OK | No error; adm_errorid = F_OK. |
| E_CLOSE_PATH/E_PARAM | Error in one of the call parameters. Either an error ID is set or the result of the access was not set by the sender module. |
| E_CLOSE_PATH/E_NOINIT | The user has not called the function tf_init before the call or the tf_init was not successful. |
| E_CLOSE_PATH/E_ASYNC | The mode = CONF_SYNC was selected, i.e. asynchronous procedure although an asynchronous job is still active. The job was not sent. The sender must wait until the end of the asynchronous job. |

### 4.4.3 Status of an Application Association

With the tf_astate_path call you can obtain information about the status of the selected application association. The reference obtained with tf_get_path_ref must be specified. The status of the application association is coded in the OPB_ADMIN parameter block.

You can choose between a synchronous or asynchronous call. In the synchronous mode (CONF_SYNC) the user process remains blocked until the call is completed.

In the asynchronous mode (CONF_ASYNC) the user process continues immediately after the call has been accepted. The result can be fetched using the tf_receive function at a later point in time. In this mode, the status of the application association is only included when it has changed.

```
INT32     tf_astate_path ( applref,
                           mode,
                           orderid,
                           ord_timeout,
                           opb_ptr
                         );

          TYP_APPLREF       applref;
          CHAR              mode;
          UNSIG32           orderid;
          UNSIG32           ord_timeout;
          struct  OPB_ADMIN *opb_ptr;


struct    OPB_ADMIN
          {
          UNSIG16           adm_errorid;              R
          UNSIG16           status;                   R
          UNSIG32           reserved;                 I
          }
```

**Function call:**

```
result = tf_astate_path ( applref, mode, orderid, ord_timeout,
                                    &opb_astate_block);
```

**Description of the parameters :**

- **Function call**

| | | |
|---|---|---|
| applref | Reference of the application association via which the job is to be sent. | |
| mode | CONF_SYNC | job with synchronous confirmation |
| | CONF_ASYNC | job with asynchronous confirmation |
| orderid | Job ID assigned by user. This ID can be used with asynchronous jobs as an option using r_orderid of the RCV_BLOCK to identify the acknowledgement. | |
| ord_timeout | Management time for the job (in seconds). | |
| *opb_ptr | Pointer to the job parameter block OPB_ADMIN. | |

- **Job parameter block OPB_ADMIN**

| | | |
|---|---|---|
| adm_errorid | Return parameter in the OPB; errors for the specific job are returned in the adm_errorid parameter. | |
| status | This return value indicates the status of the application association. The following statuses are possible: | |
| | E4V_UNKNOWN | connection not known |
| | E4V_DOWN | connection not established |
| | E4V_ESTABLISH | connection being established |
| | E4V_UP | connection is established |
| | E4V_RECOVERY | connection has broken down |
| reserved | This parameter is used internally. | |

**Values of the result :**

The following values (high word/low word) can occur :

| | |
|---|---|
| STF_OK | No error; adm_errorid = F_OK. |
| E_STATU_PATH/E_PARAM | Error in one of the call parameters. Either an error ID is set or the result of the access was not set by the sender module. |
| E_STATU_PATH/E_NOINIT | The user has not called the function tf_init before the call or the tf_init was not successful. |
| E_STATU_PATH/E_ASYNC | The mode = CONF_SYNC was selected, i.e. asynchronous procedure although an asynchronous job is still active. The job was not sent. The sender must wait until the end of the asynchronous job. |

## 4.5 Translating and Linking for MSDOS

The STF libraries for MSDOS are in the directory \sinec\stf\lib.
The names are created as follows:

<Memory model><Operating system>stf<compiler>

| | | |
|---|---|---|
| <Memory model> | l | Large model |
| | b | Big or huge model |
| <Operating system> : | d | MSDOS |
| | w | Windows |
| <compiler> : | msc | Microsoft C Compiler 6.X |
| | tc | Turbo C 2.0 or Turbo C++ 1.0 or higher |

### 4.5.1 Working with the MSC 6.0 Compiler

STF library for the MSC compiler 6.0 is \sinec\stf\lib\ldstfmsc.lib .
A test program is translated and linked as follows:

```
cl  /c /AL /Os /I\sinec\stf\include  /DM_DOS   test.c
link  test.obj,test.exe,,\sinec\stf\lib\ldstfmsc+ c:\c600\lib\llibce ,,,
```

### 4.5.2 Working with the TURBO−C Compiler

The STF library for the Turbo C Compiler 2.0 or Turbo C ++ 1.0 is \sinec\stf\lib\ldstftc2.lib .
A test program is translated and linked as follows:

```
tcc −c −ml −I\sinec\stf\include  −DTURBO_CC −DM_DOS   test.c
tlink @test.lnk
```

The instructions for the linker are in the file test.lnk:

```
\tc\lib\c0l.obj  test.obj
test.exe
test.map
\tc\lib\emu.lib  \tc\lib\mathl   \tc\lib\cl.lib \sinec\stf\lib\ldstftc.lib
```

## 4.6 Special Features of Windows

The STF library supports the **Enhanced Mode** under Windows 3.0/3.1.

The STF library must be linked to the user program. A DLL library is not currently supported.

One of the differences between Windows programs and DOS programs is that Windows programs branch to a WndProc. At a central point, Windows programs wait for Windows messages which are then processed in a WndProc procedure. It is possible that during the processing of the WndProc control is transferred to Windows and that WndProc is called again. Since the STF library is not re−entrant for a process, you must make sure that the functions of the STF library are called at the same time.

The best way of doing this is to use only the asynchronous mode.

In a Windows program, following stf_init () you must call the set_window_handle routine with a Window handle so that TF−NET1413 knows where to send its messages. If an asynchronous command is issued, a WM_SINEC message is sent to Windows when the message is received. This can then be processed in the corresponding WndProc using a tf_receive with timeout 0.

Example of a typical Windows application:

```
WndProc (hWnd, )
{
     switch (msg)
      {
            case ....  /* init –code */ :
                              stf_init ();

                               set_window_handle (hWnd);

                               break;
             case  ... /* Trigger the STF function */:
                               tf_xxx  (     ,ASYNC,   );

                                break;
             case WM_SINEC:

                        tf_receive ( 0,1, ,  );

                           break;


       }
}
```

Call format of the set_window_handle:

```
set_window_handle (hWnd)
HWND  hWnd;
```

Further examples of programs for Windows can be found in the directory \sinec\stf\sample\win.

**TF/DDE manager:**

A TF/DDE manager is implemented with the STF library and can be addressed with dynamic data exchange by a user program or a standard application such as EXCEL and handles the communication with TF−NET1413/MSDOS, Windows. This TF/DDE manager is a separate product but can only be run in conjunction with the TF−NET1413/MSDOS, Windows package.

### 4.6.1 Starting an STF Windows Application

- Start the CP 1413 driver under MSDOS.
- Start Windows.
- Start the program \sinec\bin\sin_serv.exe under Windows.
- Start your STF application.

## 4.7 Translating and Linking for WINDOWS

### 4.7.1 Working with the MSC Compiler 6.0 and the SDK from Microsoft

The STF library for the MSC Compiler 6.0 under Windows is \sinec\stf\lib\lwstfmsc.lib .
A test program is translated and linked as follows:

```
cl /c /Zi −Gw −Zp /AL /Os /I\stf\include   −DM_WINDOWS  /DM_DOS  test.c
rc −r test.rc
link /NOD   test.obj,test.exe,test.map,\stf\lib\lwstfmsc.lib+LLIBCEW+LIBW,test.def
rc −K test.res
```

NOTE : rc is the resource compiler of the Windows SDK.

❑

# NOTES

**Please replace this page with the manual of the SINEC TF User Interface**

**Register 6**
**Please replace this page with the manual of the TF/DDE – Manager**

# 7 Program Examples

The examples are stored in the directory SINEC\STF\SAMPLE. They include the most common TF−calls and show the handling of the TF− User Interface. You will find more information on the handling of the examples either in the header of the files or in the READ.ME file. There you will also find the corresponding databases.

***NOTES***

# 8 Dealing with Problems

## 8.1 DPRAM Test

Note:
If there is a problem, it may take a long time for the test to complete and the programm may appear to be ”locked up”.

Under MSDOS, the CP 1413 requires a free 64 Kbyte memory area at addresses **D0000 or E0000**. Using the program \sinec\bin\dpramtes you can check whether a DPRAM area is free for the CP 1413.

dpramtes <io_adr> <dpram_adr>

<io_adr> is the configuration address set on the hardware of the CP 1413.
3e0, 390, 3e8 or 100
<dpram_adr> is the DPRAM address to be tested.
d0000 or e0000

Example: dpramtes 3e0 d0000
checks whether D0000 is free; 3e0 is set as the configuration register on the CP 1413.

If the message
test passed
is displayed this area can be used as the DPRAM address for the CP 1413.

If you receive the message
Difference:
.....
.....
test not passed
This area **cannot** be used as the DPRAM address for the CP 1413. After the line ”difference:” the memory cells are printed out in which a different value was read from that written.

In some computers, the shadow RAM or the cache uses the area E0000. You can then only use this area for the CP 1413 when you disable the cache and shadow RAM in the system setup (cache disabled, shadow RAM disabled).

The memory area used by the CP 1413 must not be used by expanded memory managers such as EMM386, QUEM, 386MAX. When you start these managers, you must then specify that this area must not be used:

Example for EMM386 :

```
device=emm386.sys ..... X=D000-DFFF
```

This means that EMM386 will not use the area D0000 to E0000.

## 8.2 Error Codes of the Driver on the Program Interface

The driver transfers an error code to the calling program in the variable errno (type: unsigned short). If 0x9500 is entered in the high word in the results value of the STF functions, this means that the low word contains a driver error message.

The following error codes are possible:

```
#define  SCI_OK           0     /*0x0000*/   /* No error, successful                    */
#define  SCI_RESOURCE     202   /*0x00ca*/    /* The resources in the driver are exhausted*/
                                               /*                                         */
#define  SCI_CONFIG       203   /* 0x00cb */  /* Error configuring the driver            */
#define  SCI_NOCONFIG     204   /* 0x00cc */  /* The driver has not been started or      */
                                               /* is not configured                       */
#define  SCI_PARAM        206   /* 0x00ce */  /* Parameter error                         */
#define  SCI_DEVOPEN      207    /* 0x00cf */ /* The user is not (no longer) logged on   */
                                               /* Possible causes: driver not started     */
                                               /* no channel exists                       */
                                               /* Number of permitted opens exceeded      */
#define  SCI_BOARD        208   /* 0x00d0  */ /* No module exists                        */
#define  SCI_SOFTWARE     209   /* 0x00d1  */ /* The module is not responding            */
#define  SCI_MEM          210   /* 0x00d2 */  /* There is no free DPRAM page in the      */
                                               /* transmit direction                      */
                                               /* Remedy: renewed TF call                 */
                                               /* or reconfiguration of DPRAM channels    */
#define  SCI_LOADER       212   /* 0x00d4 */  /* An error occurred loading the firmware  */
#define  SCI_NOMESS       215   /* 0x00d7 */  /* There is currently no message for this  */
                                               /* process                                 */
#define  SCI_USERMEM      216   /* 0x00d8 */  /* The length of the buffer transferred to */
                                               /* driver is too small                     */
#define  SCI_TIMEOUT      219   /* 0x00db */  /* The job for the driver could not be     */
                                               /* executed in the required time           */
#define  SCI_ECLOSED      224   /* 0x00e0 */ /* The connection to the driver is already  */
                                               /* closed                                  */
#define  SCI_USERMAX      225   /* 0x00e1 */  /* The number of permitted users logged    */
                                               /* on to the driver has been exceeded      */
#define  SCI_EINTR        226   /* 0x00e2 */  /* The function was terminated with        */
                                               /* CTRL−C                                  */
```

## 8.3 Error Messages of the SCP Monitor and the Driver

### 8.3.1 Starting the Driver

Message : Error : There is already a driver on SW−INT xx

The TF−NET1413 driver uses two consecutive software interrupts. If the environment variable SCI_SW_INT is set (e.g. with set SCI_SW_INT=82 ) this software interrupt is used. If this variable does not exist, 0x80 is used as standard. If you receive the message above, a driver is already using the specified software interrupt. In this case, set the environment variable SCI_SW_INT to a free software interrupt before starting the driver.

### 8.3.2 Messages of the SCP Monitor

Message :

Driver is not loaded !

The driver tfnetdrv.exe is not loaded. Load the driver before the scp_mon command.

ERROR: couldn't open database file <file name>

During installation/configuration, you were prompted to name the data base you created with NML or COML 1413 (default : \sinec\data\startup.ldb). This name is located in the file \sinec\data\dos_conf.dat. If you receive the message above, this file does not exist. In this case, the Ethernet address specified during installation is set.

ERROR: setting ethernet address

An error has occurred setting the Ethernet address.

ERROR: couldn't initialize connection!

The initialization of an application association was unsuccessful.

ERROR: couldn't put server reference!

Logging on a server on the CP 1413 was not successful.

ERROR: Timeout: perhaps invalid interrupt number

The CP 1413 has not signalled within a specified time. Check whether the interrupt number set during the configuration is free on your computer.

ERROR: Timeout make attach : get buffer
error .....
intvector : xx
Configuration error: Please check dos_conf.dat

The CP 1413 has not signalled a hardware interrupt. Check whether the interrupt number you specified during configuration is free on your computer. Configure a different interrupt and restart your computer.

PDU − ERROR: ERRORCLS & ERRORCODE = 415c

An scp_mon −h is executed but there is no data base loaded on the CP 1413. The scp_mon −h command is executed correctly despite this message.

### 8.3.3 Error Loading the Firmware on the CP 1413

Download: Not able to open file
FAILURE IN DATATRANSFER

The file with the firmware of the CP 1413 does not exist. The file has the name sinec\data\fw1413. Reinstall TFNET 1413.

Download: Host receives no response from CP, TIMEOUT
FAILURE IN CONNECTION

The CP 1413 does not respond when you attempt to load the firmware on the module: check whether the DPRAM area (D0000 or E0000) you specified during the configuration is free on your computer. You can use the program \sinec\bin\dpramtes to check the area (see Section 8.1). If this area is not free, configure a new area.

There are jumpers on the CP 1413 with which the I/O address of the configuration register can be set. If you selected the symbol mode during configuration, the driver software works with the value 3e0. If you selected the detailed mode, the value was selected explicitly. Check whether the configured value matches the value set on the module. If the values do match and you still obtain the message above, change to a different I/O area. ❑

# 9 COML 1413 TF Introduction to Configuring the CP 1413 TF

## 9.1 Introduction

### 9.1.1 Preface

These configuring instructions describe the product COML 1413 TF. The COML configuring tool can be run on AT-compatible PGs or PCs under the operating systems MS-DOS (data base converter) or MS Windows (COML user interface). COM 1413 TF is used to configure the SINEC communications processor CP 1413 (TF-NET1413/MSDOS, Windows) for the SINEC H1/H1FO bus system.

Using the COML configuring software on a PC/PG, you can create the data base required for operating the communications processor (CP 1413). The data base contains all the selectable communications parameters and is therefore used to stipulate the communications relations (links).

For communication via ISO layer 4 (transport layer) for example with a SIMATIC S5 programmable logic controller, no configuration is required. If you want to communicate only via ISO layer 4, you can skip the section dealing with configuring.

If, however, you also want to communicate using ISO layer 7 with SINEC technological functions (TF), configuration is necessary. Here, you specify who communicates with whom (for SINEC TF). The advantages of configuration are as follows:

- When you create or install a program you do not need to know its communications partners. The communications parameters are specified during configuration.
- Greater operating reliability adapted to manufacture.

### 9.1.2 Notes on Using the Configuration Instructions

The manual is intended primarily for the person configuring an H1 communications network. The information is also important for installing and commissioning the communications processors.

A basic knowledge of handling PGs and PCs and the operating systems MS-DOS and MS Windows is assumed.

The chapters 9.2.1 and 9.2.2 provide an overview of configuration with COML or with an editor and converter.

Section 9.3 and Section 9.4 then goes into more detail about configuration with both tools.

Section 9.5 describes the textual data base also used for documentation.

Section 9.6 contains information about the SINEC NML configuration tool and about compatibility with the tools listed here.

### 9.1.3 Basic Terms

Configuration is the creation of a data base. The data base contains the configured parameters and is read in when the CP 1413 TF is started up. It can exist in two different formats, a binary and a textual format.

- The binary data base (binary DB) contains the parameters you have entered in compact binary format. It can be loaded on the CP using the SCP monitor.
- The textual data base (text DB) contains the parameters in ASCII representation. This representation can be read, modified or printed out with any editor.

The data bases contain the following information:

- The station addresses (also known as Ethernet or MAC addresses) of both partners. The station address can be considered as the “house number” of the communications card.
- The TSAPs (ISO layer 4 addresses) of both partners. This is a sub-address on the CP allowing several programs which communicate via the card to be addressed.
- Further information about the connection (for example who establishes it).
- The application association name with which the user program communicates with a partner. For the application, this name is a substitute for the communications parameters.

### 9.1.4 Installing the Software

The 3½” diskette supplied is formatted under MS-DOS (1.44 Mbytes). Installing and starting the COML software on the hard disk is described below.

▶ **Caution** ◀ Before you start installation, please remember to make a copy of your original diskette(s). Use the copy for installation.

The first part of the COML installation depends on whether you have obtained the software as a separate software package or integrated in the TF-NET1413/MSDOS, Windows package.

- **COML as separate software package:**

  ✓ Copy the content of the diskette to the hard disk. It is advisable to copy the software to the directory **SINEC****COM** (although any other directory is possible).

- **COML integrated in the TF-NET1413/MSDOS, WINDOWS package:**

  - Call the installation routine **INSTALL.BAT**. The routine is located on your first diskette. INSTALL.BAT copies the SINEC software to your hard disk to the directory **SINEC**. COML is written to the directory **SINEC****COM**.

  - The data on the diskettes is compressed and INSTALL.BAT must be used.

COML must then be installed under MS Windows. The installation consists of two steps:

- **Creating a program group** in Windows (e.g. “SINEC” program group).
  This step can be omitted if a suitable program group already exists.

- **Creating a program symbol** within the program group. The path name of the program file **COM1413T.EXE** must be entered in the text field “command line”
  (e.g. \SINEC\COM\COM1413T.EXE).

▶ **Note :** A detailed description of Windows installation can be found in your Microsoft WINDOWS user’s manual (working with groups, working with program symbols).

## 9.2 Creating Data Bases

To plan communication, a data base must be created. This is possible in three different ways:

1) by configuration using the menu-guided tool COML 1413 TF or

2) by creating a text DB with an editor and then converting it to a binary data base or

3) by using the configuration tool SINEC NML, which must be ordered separately and which allows a more complex and more detailed level of configuration.

If Windows is available, the data base should be created with COML.

If Windows is not available, a text DB can be created with an editor which is then converted to a binary form. This procedure is more difficult for inexperienced users, since several lines must often be copied and no help system is available.

Creating data bases with SINEC NML is not described here. If you have already worked with NML, you should read the last chapter which deals with the compatibility to NML.

### 9.2.1 Working with the COML 1413 TF Configuration Tool

COML 1413 TF is a configuration tool for creating a data base. The basic steps for creating a data base are shown in Fig. 1.

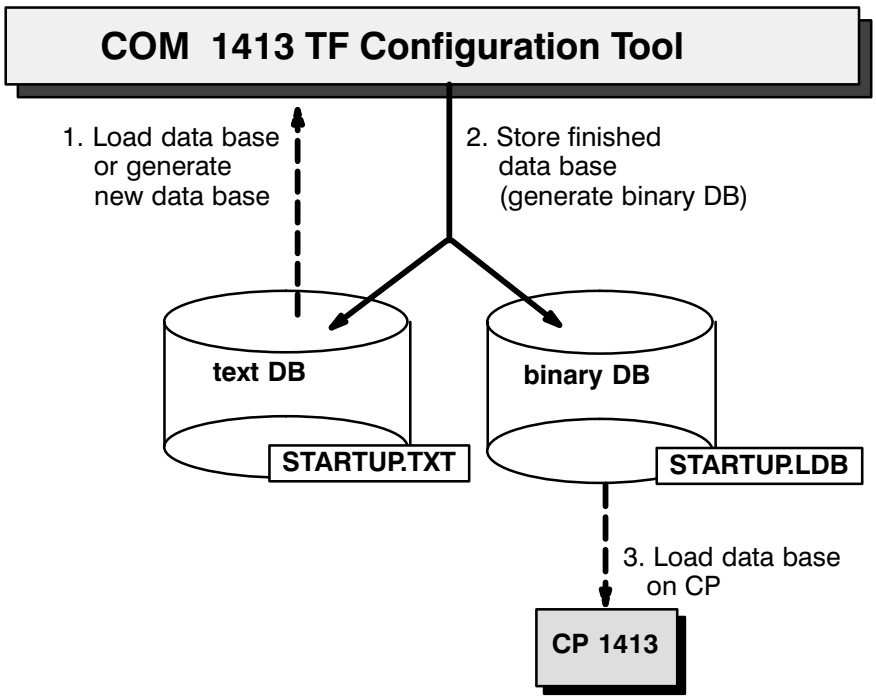



Fig. 1: Creating a data base with the COML 1413 TF configuration tool

You create a binary data base following the steps below:

- Start the Configuration tool (program com1413t.exe in directory \sinec\com).
- Create a new data base or load an existing text DB (H1.TXT) .
- Prepare the data base.
- Generate the binary DB.
- Load the binary data base on the CP using the SCP monitor.

### 9.2.2 Working with the Editor and COML 1413 TF Converters

If Windows is not available, you can generate the DB in other ways instead of using COML. To do this, you can create the textual data base (text DB) using any ASCII editor and then convert it to binary form with a converter.

The procedure for this method is shown in Fig. 2.

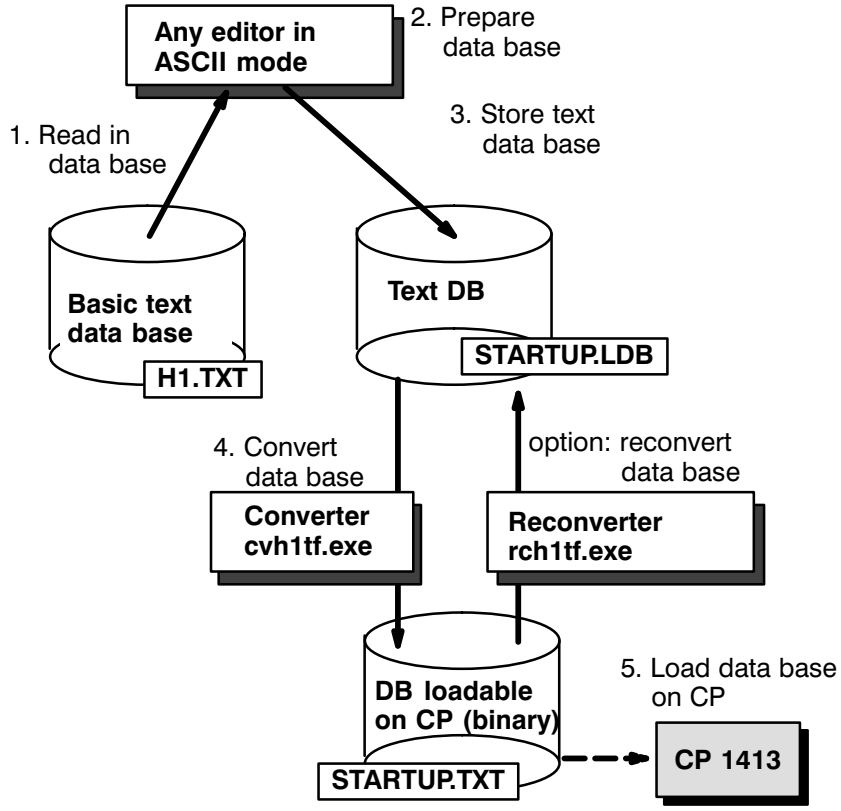



Fig. 2: Creating a binary data base using an editor and converters

To create a binary data base, follow the steps below:

- Read in the basic data base, e.g.: H1.TXT in the editor.
- Prepare the data base.
- Store the textual data base.
- Convert the text DB to a binary DB using the conversion program.
- Load the binary data base on the CP using the SCP monitor.

**Important:** The binary data base must be in the directory **SINEC****DATA**. As the name, you should select STARTUP.LDB. If you select a different name, this name must be entered in the installation tool.

## 9.3 The COML 1413 TF Configuration Tool

COML 1413 TF is a configuration tool with which you can create a binary data base under Windows. A text DB is then automatically created as well. The menu-guided user interface is more user-friendly than the converters (with an ASCII editor).

Start COML (program name "com1413t.exe" in directory \sinec\com).

COML is operated following Windows conventions. If you are not familiar with these conventions, read the following sections in the Windows user's manual:

- opening, saving files
- printing documentation
- the help system
- handling the mouse
- working without a mouse

The following sections describe the menu, the main screen form with the configuration parameters and the screen forms for working with data bases.

### 9.3.1 The Menu

### The help menu item

If you select the help item in the menu, a list with the following points is displayed. After selecting one of these points, the help you require is displayed. The points you can select are as follows:

**Index**
All the keywords for help topics are listed.

**First steps**
Contains brief instructions about the functions of SINEC COML and how to work with it.

**Using help**
Explains how to use the help functions.

**Information**
This command opens a window with general information about the program SINEC COML, e.g. Copyright and Version number.

The help system complies with the standard help system under Windows. For further information, please refer to the Windows User's Manual. Using "F1"you can obtain detailed information about the currently active element or field.

## The file menu item

If you select the file menu item with the mouse, a list containing the following points is displayed. By selecting one of these points, you obtain the required functions. The points you can select are as follows:

**New**

This command deletes all the information about the current data base. You can create a new data base.

**Open DB...** (open data base)

This command opens a window to read in a data base. You can open a text data base (*.txt) or a binary data base (*.ldb) (lower selection field). If you open a binary data base, a text data base (.txt) is generated automatically.

**Store text DB, store text DB as...** (store text data base)

This command stores the data base you are working with in textual form. With “store text DB as...”, a window is opened for you to type in a name. If you select “store text DB”, the data base is stored under its previous name.

**Generate binary DB as...**

This command opens a window to store the data base. The name must finish with “.ldb”. All input is first stored in a text file (with the file extension .txt). A check is made for errors. If no errors are found, a loadable binary data base with the extension “.ldb” is generated.

**Print documentation...**

The current data base is output for documentation on a printer. In contrast to the normal file, this documentation file has comments included. A screen form is displayed in which the printer parameters can be set.

**End**

Terminates the SINEC COML.

### 9.3.2 The Main Screen Form

The main screen form (Fig. 3) contains the data to be programmed. Some of these are specific to the CP (node name, station address, CP type) and the others are specific to the application association.

An application association describes the ISO layer 7 connection between two partners. It is identified uniquely by its name.

The main screen form contains the following:

- parameters for the specific CP (node name, station address and CP type)
- a list of the application associations (left box)

- the parameters of the current application association (selected in the list)
- the menu line for working with the file and for help
- switches to include or delete application associations.

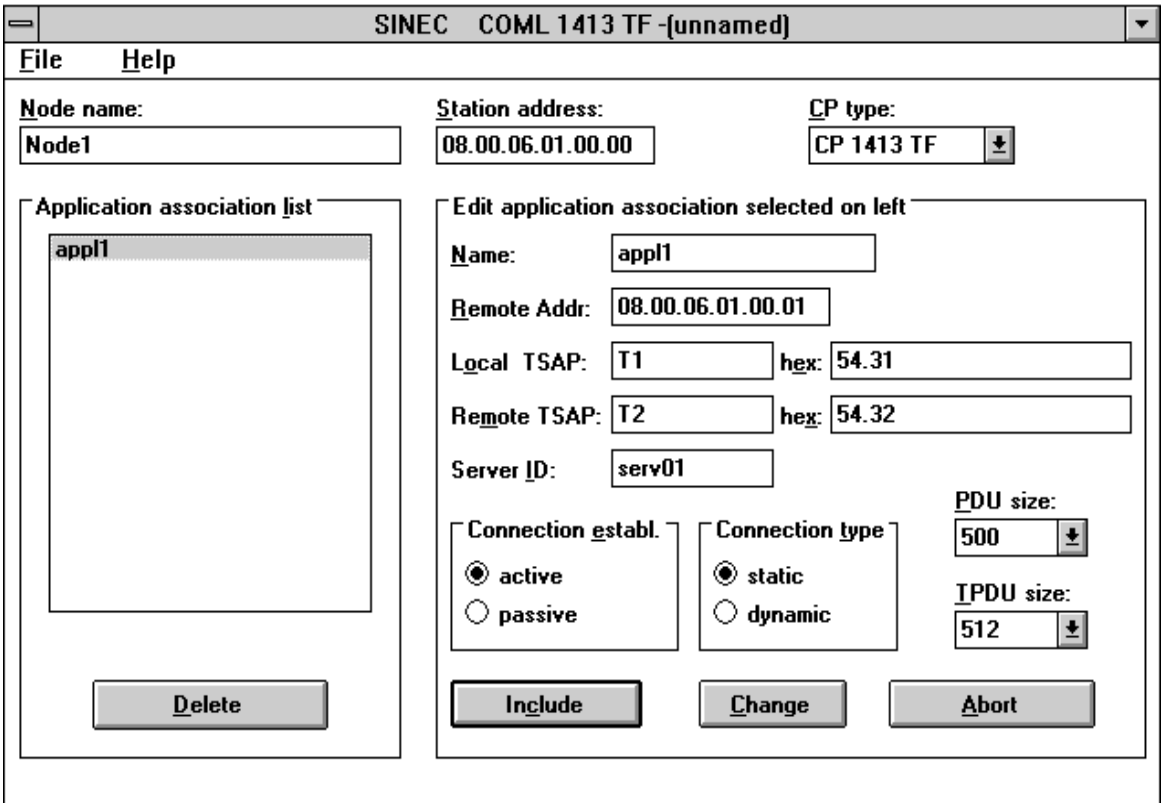



Fig. 3: Main screen form with menu and configuration parameters

The main screen form contains two large boxes, one on the left and one on the right of the screen. The left box contains the list of application associations from which you can select an application association to work with. The parameters of the selected association are displayed in the right-hand box. If you change one of these parameters, you cannot exit the right-hand box until you have acknowledged changes with the switches “include”, “change” or “abort”. This mechanism prevents accidental changes being made.

The meaning of the parameters is described in detail below. When working with COML, you can obtain help about every parameter by selecting it with the cursor and then pressing “F1”.

**Station address**

The station address is the MAC (Ethernet) address of this station. The station address must be specified. Enter six pairs of hexadecimal numbers separated by a period (”.”) or blank (” “).

Example: 08.00.06.01.00.01

Note: The first pair of numbers (08 in the example) should be even. Otherwise it is a Multicast address, which is normally assigned to a group of nodes as an additional common address.

### CP type

This entry specifies the type of communications processor. This product only supports the CP 1413 TF.

### Node name

The node name is only for user documentation and has no effect on configuration. It is a name up to 40 characters long to identify a station (this station).

### Application associations

The application associations contain the data which describe the connections between partners. The list (on the left) contains all the application associations and the box on the right displays the data of an application association.

If the list of application associations contains entries, you can display the parameters of an entry on the right-hand side by positioning the cursor on the required application association using either the mouse or the cursor keys up or down providing the list is displayed.

Using the switches “delete”, “include”, “change” and “abort” you can manipulate application associations. You can change the parameters or an application association in the right-hand box. If a field is activated on the right-hand side, the editing field for application associations can only be exited after clicking on one of the switches “include”, “change” and “abort”.

### The “delete” switch

To delete an application association, this must first be selected in the list of application associations. By clicking on the “delete” switch with the mouse, the application association is deleted.

### The “include” switch

The values input or set in the right-hand window are entered in the form of a new application association. Its name is entered in the list of application associations.

### The “change” switch

The values entered or set in the right-hand window are entered. The application association selected in the list of application associations is replaced by the new association, in other words ,the changed parameters are adopted.

### The “abort” switch

The values input or set in the right-hand window are discarded.

### The parameters of an application association

To enter an application association the parameters displayed in the “Edit application association selected on left” must be input and then entered by clicking on the “include” switch. If required, this procedure can be repeated.

The following parameters must be entered:

### Name

Symbolic name of the application association. The user program requires this name to access communications paths. The name must be specified and it can have up to 16 characters.

An application association name must only be used once on a station. All application association names are listed in the list of application associations (left box).

### Remote address

The remote address is the station address (MAC or Ethernet address) of the partner station. Enter six pairs of hexadecimal numbers separated by a period (”.”) or blank (” ”).

Example: 08.00.06.01.00.01

Note: The remote address must be specified if this station establishes the connection to the partner. Refer to the help system.

### TSAP, general description

TSAP stands for Transport Service Access Point (TSAP-ID). The TSAP must not be longer than 8 characters. The TSAP is the layer 4 address. The TSAP selected for a station and its partners must match up (remote TSAP on station A = local TSAP on station B).

TSAPs can have any characters. If non-representable ANSI characters are used, the input must be made in hexadecimal representation (field to the right beside the TSAP field).

Note: The TSAP must be specified if this station establishes the connection to the partner. Otherwise, the remote TSAP (if it is unknown) can only be specified during operation. In this case, no remote address can be specified and the connection is classed as unspecified.

Caution: The representation is in the ANSI representation, normal for Windows, and this is not identical to ASCII representation (e.g. difference with umlauts!). Special characters such as umlauts (ä, ö, ü) should therefore not be used).

With COM 143 the user inserts blanks in the TSAP name. These must also be inserted with COML 1413 TF.

**Local TSAP**

TSAP for this station (= remote TSAP of the partner station).

**Remote TSAP**

TSAP for the partner station (= local TSAP of the partner station).

**Server ID**

The server ID is used to combine several application associations together to form a logical server. A program operating as server does not need to know which application associations are connected to it. Instead, the server ID is used.

The entry is optional. Names with up to six characters can be used.

**Connection establishment**

Mode for connection establishment. The connection establishment describes who establishes the connection. With two partners, only one establishes the connection, (i.e. only one is active).

The following possibilities are available:

**Active**
This station establishes the connection (partner waits for connection establishment).

**Passive**
This station waits for connection establishment (the partner initiates it).

For partially specified, passive stations refer to the help system with the F1 key.

**Connection type**

The connection type specifies when a connection is established and terminated.

**Static** A static connection is established after start-up and maintained constantly even if no data exchange is currently taking place.

**Dynamic** A dynamic connection is established only for data exchange as required by the programs and then terminated again.

**PDU size**

Maximum size of the data buffer which the user transfers from its application to the CP. Select one of the following PDU sizes in bytes:

500
1000
2000
4000
8000

The data buffers are distributed depending on the highest value selected for an application association on the CP. If, for example, no PDU sizes of 8000 or 4000 bytes are selected, the available memory will be used for additional buffers of 2000, 1000 and 500 bytes. Whenever possible, **avoid** using **8000 byte** buffers. Recommended values: 1000 or 2000 bytes.

The PDU sizes programmed for application associations are also available for layer 4 connections (transport links) which are not assigned parameters in configuration. If no application association was programmed or only associations with 500 bytes, then 1000 byte PDUs are available for the layer 4 connections.

**TPDU size**

Size of the transport PDU (data frames via the local area network) in bytes. Possible values: 512 and 1042 bytes.

512 bytes is strongly advised. If several application associations are programmed with 1024 bytes, bottlenecks may occur since there is not enough transmit buffer space available.

The TPDU size has nothing to do with the size of the user buffers. These can be larger since the CP can distribute a user buffer on several TPDUs.

### 9.3.3 The Screen Form for Working with Files

To open or store a data base, select the required item in the file menu. The file screen forms are structured as usual in windows (Fig. 4).

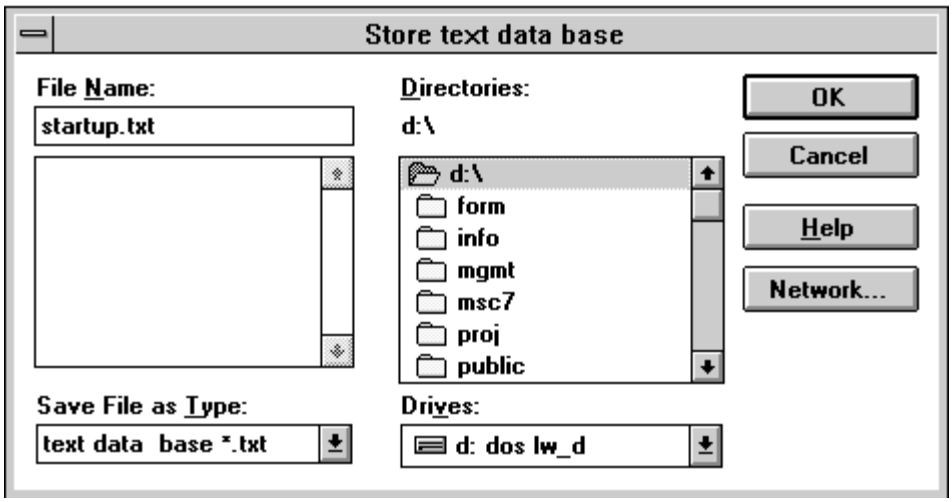



Fig. 4: Screen form for opening or storing a file

The following fields must be completed in the file screen form:

**Drives:**
Select the required drive.

**Directories:**
Select the required directory.

**List Files of Type/Save Files as Type:**
Dependent on the file type (*.txt for text data bases, *.ldb for binary data bases).

**File name:**
You select or type in a file name for the data base. If you type in the name, the extension must be “.txt” or “.ldb”.

### 9.3.4 The Screen Form for Printing Documentation

To document your data base, select the menu item “print documentation...” in the “file” menu. The results of your current work are then printed out.

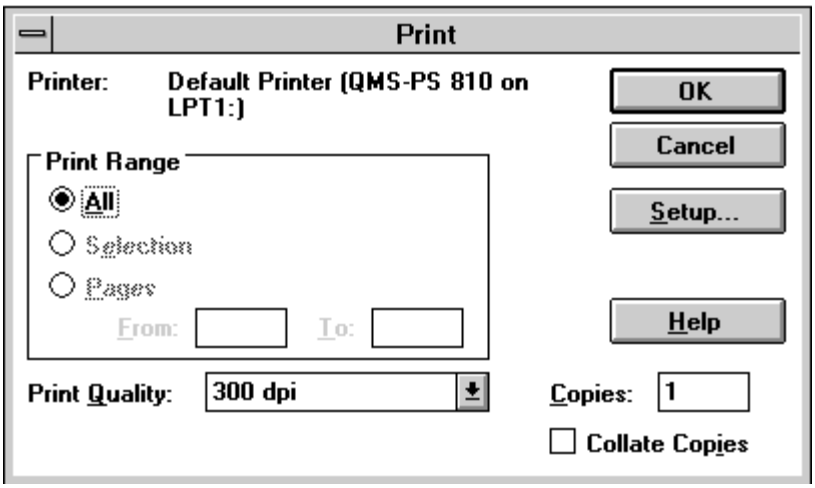



Fig. 5: Screen form for setting the parameters to print documentation.

The documentation contains all the programmed parameters with comments. The comments describe the meaning of the parameters and start with the character “#”.

**Compatibility between textual data base and documentation:**
A documentation file can be read in as a textual data base. A textual data base can also be used for documentation (abbreviated form without comments).

**Setup:**

Select the suitable printer and set the print parameters.

**Print Range:**

Select what is to be printed. Only the selection “All” is supported.

**Copies:**

Number of copies printed out (standard = 1).

### 9.3.5 The Information Screen Form

You can obtain further information about this tool such as its version by selecting the “Information” item in the “Help” menu (Fig. 6). The screen form disappears again if you click on the “OK” switch.

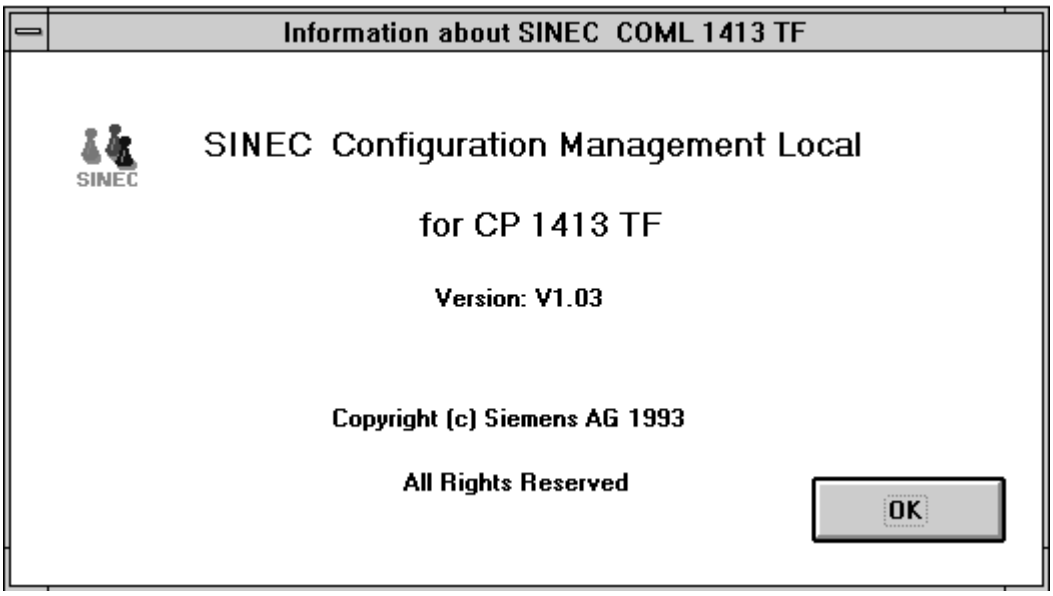



Fig. 6: The information screen form

## 9.4 The Conversion Modules and their Application

As an alternative to the COML 1413 TF configuration tool, you can use an editor to create an ASCII representation of the data base (the text DB) which is then translated into a loadable, binary form using the converter cvh 1tf.

The binary or loadable form of the data base can be retranslated to the textual representation (text DB) using the reconverter rch 1tf.

Both converters require the resource file cvh1tf.rsc.

**Calling the converter:**

cvh1tf [input file [output file]] [option]

The parameters in [ ] can be omitted. If no input file is named, h1.txt is read.

If no output file is named, h1.ldb is generated.

Option: −v for “verbose” so that after calling the converter, it outputs messages about the current processing steps.

**Calling the reconverter:**

rch1tf [input file [output file]] [option]

The parameters in [ ] can be omitted:
if no input file is named, h1.ldb is read.
if no output file is named, h1.txt is generated.

Option: −v for “verbose” so that after calling the converter, it outputs messages about the current processing steps

−s for “suppress comment”. If you select this option no comments are added to the text DB. The file is therefore much more compact.

−d for “detailed data base info”. This option has the effect that additional parameters of the local data base, that cannot be assigned parameters in the planning phase, are included as comments in the text DB.

## 9.5 ASCII Representation of the Data Base (Text DB)

The ASCII representation of the data base (text DB) consists of information and comments. Following a #, all characters up to the end of the line are interpreted as comment and ignored. In the example files, notes on using the parameters are included after the comment character.

Comments are automatically generated by the converter. User defined comments in a text DB are not supported and will be deleted by the tool.

COML 1413 TF generates a text DB without comments. The reconverter generates comments which can also be suppressed by options.

The actual information is grouped together. Each group has a name followed by a set of parameters in the form:

identifier = value.

Each group name and each assigned value is in its own line.

The following groups exist:

- **SCP2** for user documentation (exists once)
- **appl_association** for ISO layer 7 parameters (as often as necessary)

The SCP2 group must only exist once. The appl_association can occur either not at all, once or several times in a file (depending on the number of application associations).

A line with an assignment “identifier = value” always belongs to a group to which a group name is located in the previous line.

**Grammar/syntax:**

File −> group*

Group −> <group name line> value assignment line*

Value assignment line −> <identifier> = <value>

**Key to the grammar:**

| | |
|---|---|
| −> | consists of |
| * | preceding expression can occur not at all, once, or several times. |
| <Group name line> | stands at the beginning of the line. |
| <Identifier> | follows a blank or tab at the beginning of the line. Empty lines are permitted between the expressions. Comments begin with # and continue up to the end of the line. |

**Note:** A text DB with comments, as shown on the following pages, is generated in COML by printing out a data base using the function “Print documentation...” in the “File” menu, or by generating it from a binary DB using the free converter.

```
#########################################################################
#    Local Data Base for TF-NET1413 (SINEC H1)                          #
#    This file contains the textual representation of the data base #
#########################################################################
# How to use this tool:
#   -  Read the manual.
#   -  Insert the parameters you need.
#   -  Make sure the parameters match to those of the partner.
#   -  Use the program cvh1tf to generate a loadable data base.
#
# Parameters are combined into groups. One group is represented
# by a topic (e.g. association). The properties of a
# topic are described by parameters. These consist of a
# specifier, followed by = and a value (e.g. max_pdu_size = 500).
#
# Some topics can exist more than once. In this case these
# topics (key word and parameters) can be duplicated and
# modified (e.g. appl_association with parameters).
#
# Comments are preceded by #. They extend until the end of the
# line. To remove a parameter, mark it as a comment by inserting
# '#' in the first column.
#
 SCP2                    # this topic contains general parameters
                         # and is mandatory.

    local_mac_addr = 08.00.06.01.00.00 # mandatory, MAC address of this station.
                                # Enter 6 pairs of hexadecimal digits
                                # grouped in blocks of 2 and separated
                                # by '.' or ' ' (blank).

    cp_type = CP 1413 TF        # optional, currently only CP 1413 TF
                                # allowed.

    node_name = kuh             # optional, comment for documentation,
                                # name of this station,
                                # length <= 40 characters

 appl_association        # this topic contains parameters for
                         # application associations (in German:
                         # Applikationsbeziehung).
                         # For every association specify one
                         # association topic.
                         # To do this, copy a whole block and modify it.

    appl_asso_name = appl1              # mandatory, application association name,
                                # enter expression with maximum
                                # 16 characters

    appl_type = static          # optional, appl.asso. link type
                                # choose between static and dynamic,
                                # default is static.

    max_pdu_size = 2000         # mandatory, maximum size of frames in bytes
                                # choose between 500,1000,2000,4000 and 8000.

    tpdu_size = 512             # optional, TPDU - size
                                # choose between 512 and 1024.
                                # recommended value: 512
```

```
local_tsap = tsap1           # optional, local transport SAP (TSAP-ID)
                             # for this service access point
                             # (local partner of the association).
                             # Enter expression with maximum 8 characters.

remote_tsap = tsap2          # remote transport SAP (TSAP-ID)
                             # for this service access point
                             # (remote partner of the association).
                             # Enter expression with maximum 8 characters.

remote_mac_addr = 08.00.06.01.00.01 # remote MAC address
                             # enter 6 pairs of hexadecimal digits
                             # grouped in blocks of 2 and separated
                             # by '.' or ' ' (blank).

con_mode = active            # mandatory, connection link establishment mode,
                             # choose between active and passive.
                             # The active partner must initiate
                             # the establishment of a connection.

server_id = server           # optional, server identifier.
                             # It is used to group some associations
                             # to one logical server.
```

## 9.6 Compatibility with SINEC NML

SINEC NML has a much greater range of functions than the SINEC COML tool described here. If one of the following functions is required, NML should be used to create a data base.

- If function distribution tables (in SINEC AP) with several entries are used (not recommended for new applications).
- If multiplexing (in SINEC AP) is used with several application associations switched via one transport connection (not recommended for new applications).
- If parameters (e.g. transport) are to be optimized.

SINEC NML does not recognize the text DBs and cannot read binary data bases. There is therefore no way of transferring COML configuration to SINEC NML. On the other hand, COML is capable of reading the binary data bases (with the extension .ldb) from NML.

**Restrictions** using COML 1413 TF compared with NML-CP141x, CP1470:

– Communications profile:
  Only a fixed standard profile is supported for the CP. The profile is entered in a configuration file and can be modified by Siemens for a specific project (e.g. the SINEC Hotline). NML, on the other hand, provides a variety of selectable standard profiles and allows the user to configure a profile.

– Buffer pool:
  Depending on the biggest AP-PDU size (existing in the data base) a set of buffer sizes is selected automatically by COML. Five sets of buffer sizes are stored in a configuration file. With NML, the buffer pools are selected with the profiles and can be configured.

– Multiplexing more than one application association on a transport connection is not supported.

– Configuring transport connections:
  The configuration of transport connections not used by an AP application association is not supported.

– Configuring function distribution tables:
  The configuration of function distribution tables consisting of a comparison section (COMCOD, COMCLS) and a result section (server ID) is not supported. There is only one server ID per application association.

– Shortcut connections:
  Local shortcut connections are not supported.

– Remote functions:
  Functions such as download via the bus or remote diagnosis are not available. ❑

# 10 Appendix

## 10.1 Important Notes

### 10.1.1 Variable Services

With the variable services tf_read() and tf_write(), only one type of variable must be used in a call according to the variable specification. If, for example, a call contains a named variable (NAME) and a numeric variable (NUM_ADDR), this function is not executed correctly.

### 10.1.2 Remarques to Chapter 5

In chapter 5, section 10.2.4 the last component of the structure is PI_ADD_INF:

```
char    (xdoom_use) [TF_MAX_DOM_PI] [TF_MAXNAMLEN_DOM];
```

Acknowledgements sent from the TF interface, such as rsp_msg_exch, are not acknowledged extra by the module. If the user specifies incorrect values with these acknowledgements, which cannot be checked by the TF library, the function STF_OK is returned although the CP 1413 discards this acknowledgement as incorrect. For this reason, make sure that the obj_adr is correctly set with rsp_msg_exch.

## 10.2 Further Reading

/1/ NML 3.0 Configuration Software for CP 141x, CP1470 and Gracis Software with Manual
Order number : 6GK1740−0AB00−0BA0

For NML with hardware, refer to the SINEC catalog IK 10

/2/ S5 Interface Description CP 143
Order number: 6GK1970−1AA43−0AA1

## 10.3 Abbreviations, Terminology

| | |
|---|---|
| COML 1413 TF | A tool for configuring a network |
| DPRAM | Dual Port RAM : memory on the CP 1413 which is included in the AT bus of the PC and can be accessed by the PC |
| IPX/SPX | Protocols of the ISO layers 3 and 4 used by Novell Netware |
| MAP | Manufacturing Automation Protocol. |
| NDIS | Interface from Microsoft and 3Com via which the LAN MANAGER is operated |
| NML | Network Manager for LANs: a tool for configuring a network |
| ODI | Interface from NOVELL, via which NOVELL Netware and NOVELL Lite can be operated |
| PC NETWORKS | General term for all network products that can be operated by UPPS: NOVELL Netware, NOVELL Lite , LAN MANAGER, PC−NFS, SK TCP ... |
| PC−NFS | Network File System: a program package from SUN, for transparent file access by an MSDOS client to an NFS server |
| PG | Programming device |
| PG functions | |
| SCI | SINEC Communication Interface. An internal interface from TF−NET1413. |
| S5DOS−ST | S5DOS Single Tasking Stage 6: an operating system based on MSDOS for managing and configuring SIMATIC S5 program packages and communication with SIMATIC S5 programmable logic controllers. S5 DOS/ST Stage 6 is the same as STEP 5/ST2 |
| SIMATIC S5 | Family of programmable logic controllers from SIEMENS AG |
| SINEC | SIEMENS Network Architecture |
| STEP5/ST2 | STEP5/ST2 the same as S5DOS/ST Stage 6. |
| STF | SINEC technological functions |
| TCP/IP | Transport Control Protocol/ Internet Protocol |
| TF | Technological Functions |
| TIDU | Transport Interface Data Unit |
| UPPS | Universal Portable Protocol Stack: LAYER 2 interface independent of the protocol, used by adapter modules. These adapters implement both interfaces such as NDIS and ODI, and the connection to network products such as PC−NFS, SK TCP/IP, NOVELL NETWARE, LAN−MANAGER..... |

❑

# NOTES